BEHAVIORAL PROFILING OF SCADA NETWORK TRAFFIC USING MACHINE LEARNING ALGORITHMS

THESIS

Jessica R. Werling, Captain, USAF

AFIT-ENG-14-M-81

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, the Department of Defense, or the United States Government.

AFIT-ENG-14-M-81

BEHAVIORAL PROFILING OF SCADA NETWORK TRAFFIC USING MACHINE LEARNING ALGORITHMS

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Cyberspace Operations

Jessica R. Werling, B.S.C.S.

Captain, USAF

March 2014

AFIT-ENG-14-M-81

BEHAVIORAL PROFILING OF SCADA NETWORK TRAFFIC USING MACHINE LEARNING ALGORITHMS

Jessica R. Werling, B.S.C.S.
Captain, USAF

Approved:

//signed// 10 Mar 2014
Major Jonathan W. Butts, PhD (Chairman) Date

//signed// 10 Mar 2014
Michael R. Grimaila, PhD (Committee Member) Date

//signed// 10 Mar 2014
Stephen J. Dunlap, MS (Committee Member) Date

AFIT-ENG-14-M-81

Abstract

Mixed traffic networks containing both traditional information and communications technology (ICT) network traffic and supervisory control and data acquisition (SCADA) network traffic are more commonplace now due to the desire for remote control and monitoring of industrial processes. The ability to identify SCADA devices on a mixed traffic network with zero prior knowledge, such as port, protocol or IP address, is desirable since SCADA devices are communicating over corporate networks but typically use non-standard ports and proprietary protocols.

Inspired by previous research success of machine learning (ML) in accurately classifying various protocols in Internet traffic, this research uses the following four supervised ML algorithms to identify SCADA network traffic within a mixed traffic trace: Naïve Bayes, NBTree, J4.8, and BayesNet. Using packet timing, packet size and data throughput as traffic behavior categories, this research calculates 24 attributes from each device dataflow within a mixed traffic trace and introduces a novel approach of using these attributes to identify SCADA network traffic, achieving at least a .99 true positive rate (TPR). This research goes further by utilizing two attribute reduction functions to identify an optimal attribute subset, while maintaining the desired TPR of .99 for SCADA network traffic.

The attributes and ML algorithms chosen for experimentation successfully demonstrate that a TPR of .9935 for SCADA network traffic is feasible on a given network. This research also successfully identifies an optimal attribute subset, while maintaining the .99 TPR. The optimal attribute subset identified in this research provides the SCADA network traffic behaviors that most effectively differentiating them from traditional ICT network traffic.

Table of Contents

Page

List of Figures

Figure Page

List of Tables

Table Page

List of Acronyms

Acronym	Definition	
APT	advanced persistent threat	11
ARFF	Attribute-Relation File Format	33
ARP	Address Resolution Protocol	35
BAS	building automation system	66
BMS	building management system	65
CIPC	Critical Infrastructure Protection Committee	13
DCS	Dataflow Classification System	32
DHS	Department of Homeland Security	11
DPI	deep packet inspection	1
FN	False negative	41
FNR	false negative rate	41
FP	False positive	41
FPR	false positive rate	3
FTP	File Transfer Protocol	8
HMI	Human Machine Interface	7
HTTP	HyperText Transfer Protocol	8
HTTPS	HyperText Transfer Protocol Secure	36
HVAC	Heating, Ventilation, and Air Conditioning	14
ICMP	Internet Control Message Protocol	35
ICS	industrial control system	10
ICS-CERT	Industrial Control System Cyber Emergency Response Team	11
ICT	information and communications technology	1
IDS	intrusion detection system	2
IP	Internet Protocol	1
LAN	Local Area Network	1
LDAP	Lightweight Directory Access Protocol	36
ML	machine learning	1
MTU	Master Terminal Unit	7
NERC	North American Electric Reliability Corporation	13

Acronym	Definition	
NERC CIPC	North American Electric Reliability Corporation Critical Infrastructure Protection Committee	13
NIST	National Institute of Standards and Technology	11
P2P	peer-to-peer	7
PLC	Programmable Logic Controller	7
POP3	Post Office Protocol 3	15
RTU	Remote Terminal Unit	7
SCADA	supervisory control and data acquisition	1
SMB	Server Message Block	8
SMTP	Simple Mail Transfer Protocol	15
SNMP	Simple Network Management Protocol	35
SPI	shallow packet inspection	16
SUT	system under test	32
SVM	support vector machines	28
TCP	Transmission Control Protocol	15
TLS	Transport Layer Protocol	23
TN	True negative	41
TNR	true negative rate	41
TP	True positive	41
TPR	true positive rate	1
UDP	User Datagram Protocol	15
WAN	Wide Area Network	7

BEHAVIORAL PROFILING OF SCADA NETWORK TRAFFIC USING MACHINE LEARNING ALGORITHMS

I. Introduction

Internet Protocol (IP) traffic classification techniques typically rely on deep packet inspection (DPI) methods to interpret the contents of a packet's payload [37, 56]. These techniques are limited, however, for applications that use non-standard port numbers or payload encryption [37]. As an alternative to traditional classification approaches, machine learning (ML) algorithms (e.g., Naïve Bayes) have successfully used statistical network traffic attributes to classify Internet application traffic with greater than 99% accuracy [37]. This research uses four supervised ML algorithms to investigate the classification of supervisory control and data acquisition (SCADA) network traffic within a mixed network traffic trace with a desired true positive rate (TPR) of at least .99. This chapter introduces the problem, research goals and approach, as well as the assumptions and limitations of the research.

1.1 Problem Overview

It is now common for SCADA network traffic to be integrated into corporate Local Area Networks (LANs) creating mixed networks containing both SCADA and traditional information and communications technology (ICT) traffic [1]. Corporations in the process industry (e.g., electrical, water treatment, and manufacturing) are assigning IP addresses to SCADA devices to provide remote control for technicians and monitoring capabilities for usage reports and billing information; consequently, these devices may now be accessible from the Internet.

A 2012 Project SHINE [7] report showed that researchers have identified over 1,000,000 unique IP addresses of SCADA associated devices connected to the Internet. Many times the SCADA device owners are unaware that their devices are connected to the Internet [42, 48]. Other times, default configurations are set to check for software updates via the Internet. The need for situational awareness techniques for identifying SCADA devices on mixed traffic networks is increasingly important [11, 48]. Traditional methods of device identification that utilize port number, protocol, and IP address are useful; however, non-traditional methods such as using flow-based statistics are necessary when this information is insufficient or unavailable.

Previous research has demonstrated the ability to classify traditional ICT network traffic using statistical traffic attributes [2, 9, 31]. The research has aided in the creation of anomaly-based intrusion detection systems (IDSs), which use behavioral traffic patterns and protocol signatures to detect network traffic anomalies [9]. Little research has been done, however, specifically using behavioral analysis to characterize SCADA network traffic thus far. The ability to identify SCADA network traffic on a mixed traffic network without prior knowledge of protocol, port, or IP address is necessary as SCADA devices tend to use proprietary protocols and non-standard ports [48].

Twenty-four statistical flow-based attributes are calculated from each dataflow (i.e., device-to-device communication) in the mixed network traffic trace containing both SCADA and traditional ICT device dataflows. The 24 attributes are characterized by three categories of traffic behavior: packet timing, packet size, and data throughput. The categories are selected based on the known attributes of many SCADA protocols which are: deterministic, hierarchical and consistent due to their polling nature [1, 2]. The categories and attributes selected are expected to differentiate SCADA network traffic from traditional ICT network traffic.

This research also aims to identify an optimal subset of attributes from the full 24 flow-based attribute set that identify SCADA network traffic with a minimal decrease in TPR. Determining an optimal set of SCADA network traffic attributes can help create a network traffic classification device specifically designed for SCADA networks, as well as assist with the identification of typical SCADA device dataflow characteristics.

1.2 Research Goals

This research proposes that supervised ML algorithms will successfully identify SCADA network traffic within a mixed traffic trace containing dataflows from both SCADA and traditional ICT devices. There are two main goals for this research. The first goal is to identify SCADA network traffic within a mixed network traffic trace using flow-based attributes based on packet timing, packet size and data throughput with a desired TPR of at least .99. The second goal is to identify an optimal subset of attributes while maintaining at least a .99 TPR for SCADA network traffic classification.

1.3 Approach

The mixed traffic dataset used in this research is formed using traffic collected from a real-world oil and gas company network, a water and waste water treatment facility network, and a traditional ICT network. Four supervised ML algorithms: Naïve Bayes, Naïve Bayes Tree (NBTree), Bayesian Network (BayesNet), and J4.8 Decision Tree, are evaluated using the TPR and false positive rate (FPR) over three different attribute subsets. Classification model build time and dataset classification time are also examined to determine implementation feasibility in a real-world traffic classification device.

The Weka ML toolkit, used in this research, is an open source application that provides a collection of ML algorithms and tools for data pre-processing, classification, regression, clustering, association and visualization [19]. Weka contains two attribute reduction functions, the wrapper function and the filter function, to use with classification algorithms. The wrapper function is used to identify an optimal attribute subset from the full attribute set, with minimal loss of classification accuracy, depending on the classification algorithm used [44]. The filter function is used to rank all attributes in the full attribute set from most to least effective for classifying a dataset [45].

1.4 Assumptions and Limitations

In this research two assumptions are made about the collected network traces. The first assumption is that each 24-hour trace collected represents the network activity of a typical day for the SCADA and traditional ICT networks. The second assumption is that the SCADA and ICT networks used to collect the traces represent typical networks of their respective type (i.e., the ICT network used in this research possesses network traffic behavior common to most ICT networks).

There are two limitation of the research. The first is that a real-world mixed traffic network, containing both SCADA and traditional ICT traffic, was unavailable to obtain a traffic trace from. The experimental dataset is created by combining dataflows from traffic collected on real-world SCADA and traditional ICT networks to create a mixed network traffic trace. The second limitation is that both SCADA networks, where the traces were collected, use the Modbus protocol, therefore, Modbus is the only SCADA protocol in the final dataset. Due to the research limitations, the experimental results may not extend to different SCADA protocols or networks.

1.5 Thesis Overview

Chapter 2 describes a typical SCADA network, current threats to SCADA devices, the basics of network traffic analysis and behavior characterization and ML techniques used in solving classification problems. Chapter 3 details the experimental setup, dataset collection and creation methods, and performance metrics of the ML algorithms. Chapter

4 presents the results and analysis of the experiments. Chapter 5 provides conclusions of the research and suggests areas of future work.

II. Background

This chapter presents an overview, background information and related research. Section 2.1 provides basic information about SCADA systems. Section 2.2 discusses security threats to SCADA systems. Section 2.3 examines the current approaches to network traffic analysis and examines previous research performed to characterize traffic behavior for both SCADA and traditional ICT networks. Section 2.4 describes ML and supervised-learning algorithms.

2.1 Supervisory Control and Data Acquisition (SCADA) System Overview

SCADA systems control and monitor processes for water distribution, oil and natural gas pipelines, electrical utility transmission and distribution, and other systems that provide a critical public service [48]. SCADA systems are comprised of three levels: a management level, a communication transport level, and a field level. Figure 2.1 shows an overview of the three basic levels of a SCADA system.

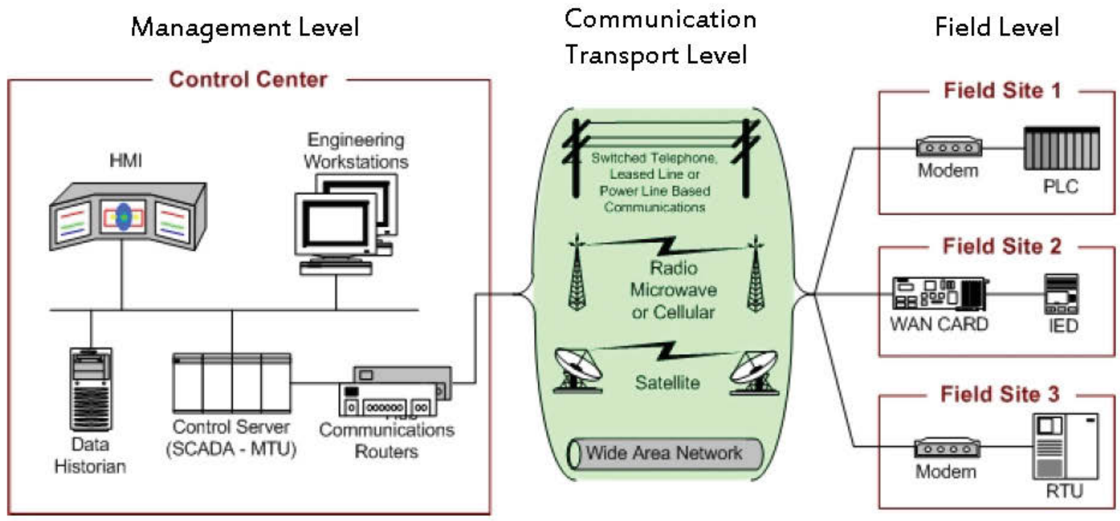


Figure 2.1: SCADA System Levels [48].

The management level is the system control center containing the Human Machine Interface (HMI), data historian server, and engineering workstations, which are all connected by a LAN [48]. SCADA system operators use the HMI to remotely control and monitor field device operations [6]. System operators also handle centralized system alarms, trend analyses, and reporting from the field devices at the management level [48].

The field level consists of one or more field sites containing field devices, such as Programmable Logic Controllers (PLCs) and Remote Terminal Units (RTUs), that control the local process actuators and monitoring sensors [48]. The management and field levels are connected via the communication transport level. The communication transport level is the Wide Area Network (WAN) providing remote access capability to operators in the control center. Field device information is transported between the control center and field sites using a variety of communication techniques found at various SCADA facilities such as telephone line, cable, fiber, or radio frequency.

SCADA and traditional ICT networks serve different purposes; therefore, have unique hardware and software characteristics that set them apart. Typical hardware on a SCADA network includes a Master Terminal Unit (MTU) in the control center, communication equipment such as a radio or cable, and field sites consisting of PLCs and RTUs to control local processes [48]. PLCs and RTUs are embedded devices configured to perform specific process tasks and generally handle a range of inputs and outputs generated by the local process sensors and actuators. Furthermore, SCADA networks have limited user interaction compared to ICT networks.

SCADA network communication is typically hierarchical in nature where a master device polls many field devices to obtain status information (e.g., temperature reading) or to send control commands (e.g., close a valve) [6]. Traditional ICT networks, however, are peer-to-peer (P2P) in nature and do not have strict timing requirements that necessitate device polling.

SCADA networks exhibit more consistent behavior in both topology and periodicity of packets sent between devices [6]. End devices are not added or removed as often in SCADA networks as in traditional ICT networks, creating a more static topology. Polling intervals of SCADA master devices exhibit periodical behavior for packets sent between devices, unlike traditional ICT networks where human interaction affects sent packet intervals.

Application protocols associated with traditional ICT networks typically use standard public communications protocols such as HyperText Transfer Protocol (HTTP), File Transfer Protocol (FTP), and Server Message Block (SMB). However, many protocols used on SCADA systems are proprietary [48]. SCADA protocols tend to be deterministic in nature such that given an input the device output is predictable [1, 2]. Standard ICT application protocols can be non-deterministic, in that they are required to handle various user-generated inputs and provide multiple outputs based on those inputs.

Utilizing SCADA systems to automate large or distributed industrial processes has many benefits including a reduction in operational costs, maintenance and overall safety [48]. A corporation creates a mixed traffic network when they connect SCADA systems by assigning the devices an IP address on their traditional corporate LANs. There are many reasons corporations connect their SCADA devices to their corporate LAN, namely, to provide remote control and monitoring for technicians and engineers, to collect production and usage data for billing purposes and to report energy usage for government green efficiency efforts. Many times the SCADA device owners are unaware that their devices are connected to the Internet [42, 48].

Default configurations on devices are often set to check for software updates via the Internet. Indeed, as mixed traffic networks in the corporate environment become more common, the ability to distinguish between SCADA device traffic and traditional ICT device traffic is increasingly important [11, 48]. Figure 2.2 provides an example mixed traffic network configuration. In the figure, the SCADA network is accessible via the corporate LAN and separated from the Internet by two firewalls.

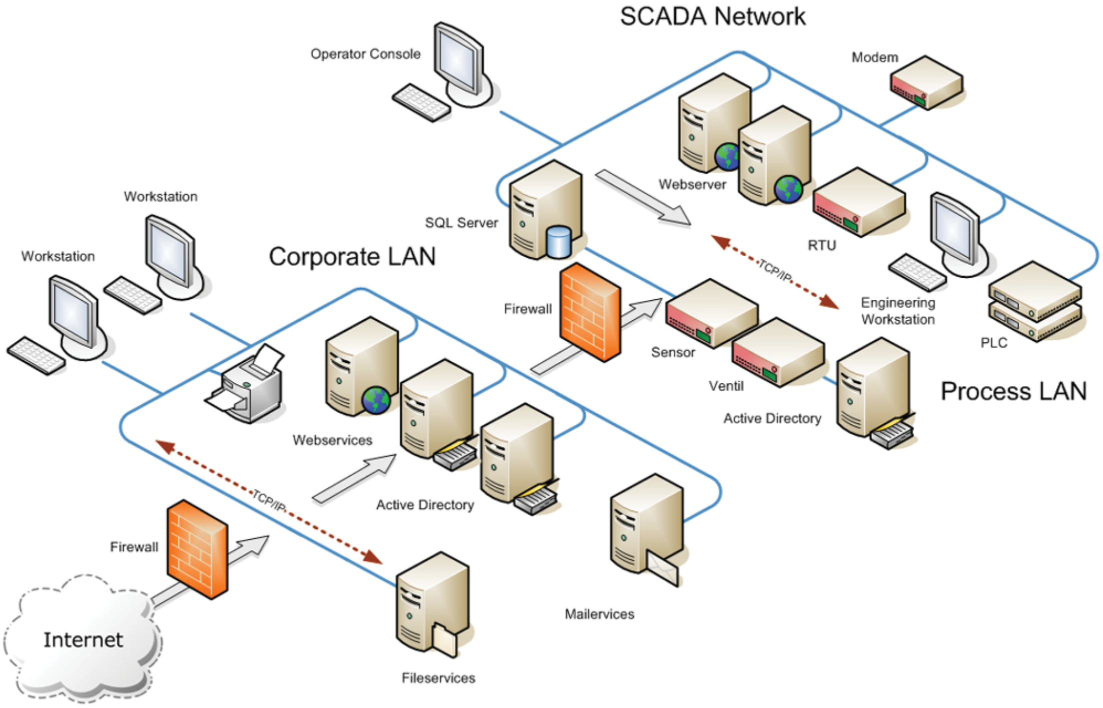


Figure 2.2: Mixed Traffic Network Configuration [17].

2.2 Threats to SCADA Systems

2.2.1 Previous SCADA Attacks.

Threats to SCADA systems can be intentional, unintentional, targeted or non-targeted, and can come from a variety of sources [41]. Shodan is a search engine for Internet connected devices. It has been referred to as “Google for Hackers” due to the ease in which it identifies attack targets, including Internet-connected SCADA devices [35]. According to ICS-CERT [22], “Hackers are using the Shodan computer search engine to find Internet-facing ICS systems” which represent potentially insecure mechanisms for authentication and authorization.

An example of a hacker penetrating network security at a Harrisburg, Pennsylvania water filtering facility occurred in October 2006 [14]. Operating over the Internet, the attacker compromised an employee's laptop and used its remote access to the facility's SCADA system to install a virus and spyware. The attack was discovered before damage occurred, however, if successful, the compromised SCADA system could have been used to distribute emails and pirated software.

In August 2006, an insider attack on a SCADA system occured when two Los Angeles city employees hacked into the system that controlled the city's traffic lights [41]. They disrupted the traffic signals at four intersections causing backups and delays. The attack was launched prior to an anticipated labor protest by city employees [21].

A recent cyber intrusion against a critical system was reported in May 2013 when U.S. intelligence agencies traced an intrusion into the U.S. Army Corps of Engineers National Inventory of Dams (NID) to a suspicious IP address [16]. The database contained sensitive information on vulnerabilities of the 8,100 major dams across the U.S.[10]. The NID included information such as the number of people that would be killed if a dam were to fail. During the attack, an unauthorized user gained access to the sensitive information without the proper clearance. The unauthorized access was revoked once discovered; however, the user had access to the NID for about three months.

In 2010, Stuxnet, a well known industrial control system (ICS) attack, targeted PLCs controlling ICS processes in a nuclear facility [15]. The worm injected code into the PLC that altered the system's motor speed, causing damage to the components. Stuxnet propagated to a closed ICS network using a removable media device. It successfully hid its malicious activities, destroyed the ICS components and demonstrated the capability of achieving significant physical effects using a highly targeted attack.

2.2.2 SCADA System Attack Techniques.

SCADA systems are becoming less isolated and more vulnerable to security threats. According to a National Institute of Standards and Technology (NIST) recently published Guide to ICS Security, SCADA technology is advancing by integrating traditional ICT system solutions like IP-based communications and standard computers [48]. The cyber-physical nature of SCADA devices make them a target for malicious attack since their compromise can lead to human safety issues, environmental issues, and critical service outages.

According to the Industrial Control System Cyber Emergency Response Team (ICS-CERT) Monthly Monitor newsletter [21], in the first half of the fiscal year 2013, ICS-CERT responded to over 200 cyber incidents across all of the 16 Department of Homeland Security (DHS) identified critical infrastructure sectors. Of these, 53% were in the energy sector. The incident responses represent a variety of threats ranging from advanced persistent threats (APTs) to sophisticated and common malware found in the SCADA environment. Other incidents in the water and commercial ICS sectors involved Internet-facing systems with weak or default credentials. The attack techniques used in the majority of incidents were: Watering hole, SQL injection, and Spear phishing attacks. All three attack techniques utilized the Internet to gain access to a connected SCADA system.

Watering hole attacks follow four main steps [50]. First, an attacker profiles a victim and learns the type of websites the person frequently visits. Next, the attacker tests all of the websites for security vulnerabilities. When the attacker finds a vulnerable website, they inject JavaScript or HTML code to redirect visitors to a new website hosting malicious code. Lastly, the compromised website waits for a victim to visit and infect them with zero-day exploits, similar to a lion waiting at a watering hole.

SQL stands for structured language query and is the primary programming language used to manage data within a database [38]. Database queries performed over the Internet primarily use SQL to access a server's database to grant system access, obtain product or account information or access data. According to the Open Web Application Security Project (OWASP), a SQL injection is a type of attack where a maliciously structured SQL query is inserted as the input data from the client side to the application database. A successful attack can access sensitive data from a backend database, modify existing data within the database and execute administrative commands on the database by allowing the attacker to spoof their identity.

Phishing is a type of cyber attack using fraudulent emails to trick victims into giving an attacker personal or financial information [51]. The emails appear to be from a legitimate website where a user may have an account such as PayPal or BestBuy. Spear-phishing is a more targeted version of phishing where the email appears to come from an individual or business the individual may know. While phishing attacks are more general and usually sent out as spam to as many victims as possible, spear-phishing is specifically directed and tailored to an individual. This type of attack uses personalization tactics such as using the victim's first name or a recent online purchase to gain trust, tricking them into providing sensitive information such as account numbers, passwords or financial information.

2.2.3 Mitigation Efforts and Recommendations.

2.2.3.1 Organizational Efforts.

The U.S. Department of Homeland Security (DHS) is a government organization charged with securing the nation from threats [55]. DHS's Office of Infrastructure Protection works with public and private sector critical infrastructure partners and leads a coordinated national effort to mitigate risks to the nation's critical infrastructure through the development and implementation of a protection program. The Homeland Security Presidential Directive 7 (HSPD-7) [53] and the National Infrastructure Protection Plan (NIPP) [54] provide the overarching framework for a partnership between the government and private sectors for the protection of critical infrastructure [55]. These documents serve to establish national policy in order to identify, prioritize and protect the nation's critical infrastructure.

The ICS-CERT is also part of DHS and coordinates with other organizations to reduce risks within and across all critical infrastructure sectors by partnering with law enforcement, intelligence agencies and federal, public and private control system owners, operators and vendors [22]. They collaborate with international and private sector Cyber Emergency Response Teams (CERTs) to share ICS-related security incidents and mitigation measures. They publish alerts, advisories and quarterly newsletters to keep the ICS community informed of newly discovered vulnerabilities and also provide teams of cyber attack experts as incident responders for federal, public and private ICS owners and operators.

The North American Electric Reliability Corporation (NERC) is a not-for-profit entity whose mission is to ensure the reliability of Bulk-Power Systems in North America and provide assurance to the public, private and government for the reliable performance of electric systems [36]. The NERC's Critical Infrastructure Protection Committee (CIPC) was formed to advance the physical and cyber security of the critical electricity infrastructure of North America. The North American Electric Reliability Corporation Critical Infrastructure Protection Committee (NERC CIPC) works closely with organizations responsible for cyber security in all electric industry segments and the government. They assist in the development of critical infrastructure protection standards and conduct forums and workshops for educating those involved in the protection of critical infrastructure and prevention of cyber attack incidents [36].

2.2.3.2 Recommendations.

The President's Critical Infrastructure Protection Board within the Department of Energy has published 21 best practices for securing SCADA systems [11]. The first three recommendations are to identify all connections to the SCADA network, disconnect unnecessary connections to the SCADA network, and strengthen the security of all necessary connections to the SCADA network. These three recommendations highlight the need for situational awareness of all connections to a SCADA network (e.g., internal LAN, WAN, the Internet, business partners, vendors or regulatory agencies) as a first step in securing the network. Other recommendations identify the need for system isolation such as removing unnecessary network services, implementing internal and external IDS systems, and conducting physical security surveys of remote sites.

Eliminating possible backdoor entry into a SCADA network can be a daunting task for mixed traffic networks which include both SCADA and traditional ICT devices. Despite security organization recommendations to establish situational awareness and isolation of SCADA networks, there is a steady increase in SCADA devices connected to the Internet. Project SHINE (SHodan INtelligence Extraction) is a project developed to extract information about the existence of SCADA and ICS devices accessible from the Internet [7]. Project SHINE uses Shodan to look for device service banners to determine device types such as computers, printers, switches, PLCs, RTUs, and anything else with an IP address. Project SHINE has been collecting data since mid-April 2012 and reports that the average number of new SCADA associated devices found every day is typically between 2000 and 8000. To date they have collected over 1,000,000 unique IP addresses that appear to belong to traditional SCADA devices such as PLCs, RTUs, and HMI servers, and non-traditional SCADA devices such as Heating, Ventilation, and Air Conditioning (HVAC) or environmental control systems, traffic light control systems, and medical devices. Many times the SCADA system owner is unaware that their devices are connected to the Internet, highlighting the need for SCADA device identification [42, 48].

2.3 Network Traffic Analysis and Behavior Characterization

2.3.1 Introduction.

Traffic analysis of transport layer applications, such as Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) packet inspection, is an essential part of an effective network defense strategy. Common analysis tools such as IDS applications, Wireshark, and Tcpdump, provide various means to view, analyze, and compare traffic to known malicious signatures and anomalies. These capabilities provide situational awareness of network activity and possible early detection of attacks.

Cheung *et al.* use model-based techniques for analyzing SCADA network traffic. They describe model-based approaches as more feasible for detecting new, unknown dataflows on SCADA networks than traditional ICT networks since SCADA networks tend to have a static topology, regular traffic patterns, and a limited number of protocols and applications [9]. In their research, they utilize the open-source signature-based IDS, Snort, and specify misuse and attack rules, based on device endpoints (e.g., IP addresses and port numbers) and other packet attributes (e.g., keywords in the packet payload). They develop models to characterize the expected behavior based on Modbus\TCP specification and create Snort rules to detect the complement of their models.

While the approach of using the Snort IDS to detect network traffic violations was successful, there are limitations. One limitation is that the customized rules required to use Snort as a model-based IDS necessitate hand-coded, protocol-specific rules [9]. Another limitation is when the communication protocols are unknown, such as when a proprietary SCADA device is added to a network, the protocol-specific rules could miss an intrusion violation.

It is also typical for traditional ICT network devices to communicate using known or assigned port numbers and public-domain protocols such as HTTP, Post Office Protocol 3 (POP3), or Simple Mail Transfer Protocol (SMTP). However, with SCADA devices, the ports and protocols used can be non-standard and propriety, making them difficult to identify when analyzing traffic [1, 37]. A large corporate network contains numerous interconnected devices, which can include SCADA devices, running various applications and protocols that communicate over the same infrastructure. As end devices are added and removed from a network, it becomes increasingly difficult to maintain full situational awareness of every device connected at a given time.

2.3.2 Methods for Analyzing Network Traffic.

In-depth analysis of network traffic is critical for network defense. Tools for analysis come in the form of hardware containing specialized software or software installed directly onto a host computer. A variety of free network traffic analysis software is available such as Wireshark [58] and Tcpdump [52], as well as commercially available tools. These tools collect dataflows traversing a selected network interface and decode the data packets of known protocols, displaying them in human-readable format [58]. Most traffic analysis tools perform a combination of both shallow packet inspection (SPI) and DPI on packets of known protocols. In SPI, the tool only inspects the header portion of a packet to include source and destination port number, source and destination IP address, and transport layer protocol, such as TCP or UDP [8]. SPI does not look at the packet's application layer protocol or payload.

Some protocols are associated with known ports such as HTTP on port 80, HTTPS on port 443, and SMTP on port 25. A web or email server is readily identified by port number using SPI. In other cases, the knowledge of the port number or protocol for a device is not available. As a result, SPI is not enough to identify a device or dataflow. DPI can assist in the identification process by examining the entire packet, including application protocol and payload content [8]. Traffic analysis tools that perform DPI must decode the application's protocol in order to display the payload content. This can be an issue with proprietary protocols, such as those used in many SCADA devices, as the protocol specifics are likely unknown. The information obtained using SPI and DPI is helpful in most cases, but is not enough to identify all devices or dataflows on a network.

2.3.3 Previous Traffic Analysis Research.

Significant research has been accomplished in traditional ICT network traffic behavior characterization. According to Moore *et al.* [34], timing attributes can be used to differentiate between dataflows during analysis. For HTTP "80% of HTTP requests occur within three seconds of each other and 95% occurs within a minute and a half" [4]. If an HTTP dataflow does not follow this pattern, it is considered abnormal since HTTP traffic occurs in short bursts as pages load, with timing gaps while the end nodes receive the data. Such known behavior can be used to identify an HTTP dataflow with reasonable accuracy, even if there is no other identifying information.

SCADA device traffic differs from traditional ICT device traffic. SCADA networks are typically static, both in terms of network topology and tasks performed [9]. Servers and hardware are not frequently added to a SCADA system once it is operational and it typically only has one purpose: running the field devices and ensuring the proper operation of critical processes. SCADA networks are also regimented in how data from remote hosts and commands are distributed throughout the network. Packets in a SCADA network are typically generated in a polling fashion where a master requests information from a number of slave devices. However, there are instances where the slave devices initiate dataflows to notify the master of an issue. Furthermore, SCADA protocols generally lack authentication and encryption due to operating requirements and use of antiquated devices.

The number of services or protocols on a SCADA network are usually limited; however, it is now commonplace for SCADA device traffic to connect to traditional ICT networks, which contain common services (protocols) (e.g., HTTP, VoIP and instant messaging) [1].

Barbosa *et al.* [1] note that previous research shows a self-similar nature of traditional ICT network traffic. The presence of long-range dependencies and heavy-tailed distributions of packets has lead to traffic models and tools for optimizing network design and management. In their research, they set out to verify if the traffic models used to describe traditional network traffic can be applied to SCADA device traffic. They introduce a number of ways to compare network behavior from the attributes of self-similarity, topological properties and application specific aspects. In their research they used four SCADA network traces and an ICT network trace from an educational organization. They test the following four attributes present in both traces for similarity:

- **Diurnal Patterns of Activity**: Network activity which correlates with human activity (e.g., work and lunch).
- **Self-Similarity**: The network trace as a whole resembles parts of itself (e.g., burstiness patterns).
- **Log-normal Connection Sizes**: When typical connection size of a trace is plotted, its curvature is log-normal.
- **Heavy-tailed Distribution**: File sizes, transfer times and burst length, when plotted, show heavy or long-tailed distribution in their curves.

The results show that SCADA network traffic does not display the diurnal patterns of activity present in regular ICT dataflows. SCADA traffic has a time series that remains stable over large periods of time, whereas regular ICT traffic has lower throughput during the evenings and weekends and peak throughput during regular business hours, in particular at the start of the day and during lunch hours. The self-similarity analysis performed using packets per second and bytes per second time series showed that the ICT traffic presented self-similar behavior, while the SCADA datasets indicate a non-self-similar behavior.

When observing the behavior of dataflow size, the ICT datasets illustrated both heavy-tailed for packets per flow and log-normal distribution for flow duration; however, the SCADA datasets were not always conclusive - one dataset had a tail distribution similar to heavy-tailed while others were skewed in a variety of ways [1]. This implies that different SCADA networks do not display similar file sizes, transfer times or burst lengths, which traditional ICT networks share a commonality when plotted. They used real-world SCADA traffic traces to perform their research and concluded that network traffic models used to describe traditional ICT network traffic cannot be applied to SCADA network traffic [1].

In related research, using the relatively static nature of SCADA topoplogy, Mahmood *et al.* [31] perform flow-based network traffic analysis on SCADA networks by observing significant changes in the number of dataflows present on the network to detect anomalies. They also capitalize on the fact that SCADA data packets are typically generated in a polling fashion where a master device polls a number of slave devices for data [31]. Having only knowledge of the number of device dataflows present during normal operations and flow rate patterns among the devices on the network they are able to successfully detect probing attacks, malware propagation, rogue master or slave devices and flooding-based denial-of-service attacks.

While this approach was successful on a SCADA network, it does not translate to a mixed-traffic network containing both SCADA and traditional ICT devices. A mixed-traffic network is less static than a SCADA network due to devices being added and removed on a regular basis as well as having less predictable packet timing characteristics. Traditional ICT devices do not generate packets in a periodically consistent manner like master devices polling slave devices in a SCADA network because their packet timing is influenced by human activity and much less predictable.

2.4 Machine Learning (ML)

Most network traffic analysis techniques rely on the 5-tuple packet information of source and destination IP address, source and destination port number, and application protocol (e.g., [9]). While this method works well for applications that use known protocols and ports, it has limitations for SCADA devices that may use proprietary protocols and unknown ports. ML offers an alternative to traffic classification with a number of algorithms demonstrating a high accuracy (i.e., up to 99%) for a wide range of Internet application traffic [37]. This section describes ML and how it can be used to classify network traffic using flow-based statistical attributes of a dataflow, rather than information found within the packet.

2.4.1 Types of ML Techniques.

ML is historically defined as a collection of powerful techniques or algorithms for data mining and knowledge discovery which search for useful patterns in data [37]. Internet traffic classification approaches apply ML techniques to recognize statistical patterns in observable attributes of network traffic such as flow duration, packet size, and inter-packet arrival time to determine the source of a dataflow [37]. Previous ML traffic classification research demonstrates the ability to observe distinctive dataflow attributes for a number of TCP applications [10, 12, 26, 27, 40].

ML is implemented into facial recognition software [18], spam detection software [39], and assists financial institutions with credit card fraud detection [32] without the need for human interaction. Stuart *et al.* [49] outline the three main types of ML: supervised, unsupervised, and reinforcement.

In supervised ML algorithms, a classification model is generated which maps inputs to outputs [49]. Figure 2.3 provides a visual representation of the supervised learning process. Data classification using supervised ML is a two-step process [20]. The first step is the learning or training phase, where the algorithm is provided a training dataset containing labeled data samples with attributes. The labels are the classes in which each data sample belongs. The knowledge gained during the training phase can be presented as a flowchart, decision tree, rule set, or other model that is later used to classify new unlabeled data samples [37]. The second step is the testing phase, which utilizes the model built during the training phase to perform classification on a new unlabeled dataset [37], labeling each sample with the predicted class to which it belongs. A number of supervised learning algorithms exist, each differing in the way their classification model is constructed and what search algorithms they use.

Unsupervised ML algorithms, also referred to as association learning, perform classification in two phases as well; however, the training component is given no hints about the structure of the data inputs (e.g., the labels in supervised learning) [49]. The training phase builds a classification model based on patterns or associations it detects between the samples in the given dataset. Clustering is the most common unsupervised learning technique. Clustering algorithms naturally discover groups using internalized heuristics and focus on finding patterns in the given dataset [37]. Hierarchical and Simple K-means are two examples of clustering algorithms [60].

Reinforcement ML is most prevalent in the artificial intelligence community. Reinforcement learning creates intelligent agents through a system of rewards and punishments without specifying how the task will be accomplished [24]. This learning model typically consists of a discrete set of environment states, a discrete set of agent actions and a set of reinforcement signals, typically 0 or 1 [24]. The intelligent agent's goal is to gain the largest sum of reinforcement signals (i.e., rewards), which it acquires after choosing an action with a reward attached [24].

2.4.2 Supervised ML Algorithms.

A number of supervised ML algorithms exist for data classification, each differing in the way they construct the final classification model. Four supervised ML algorithms are

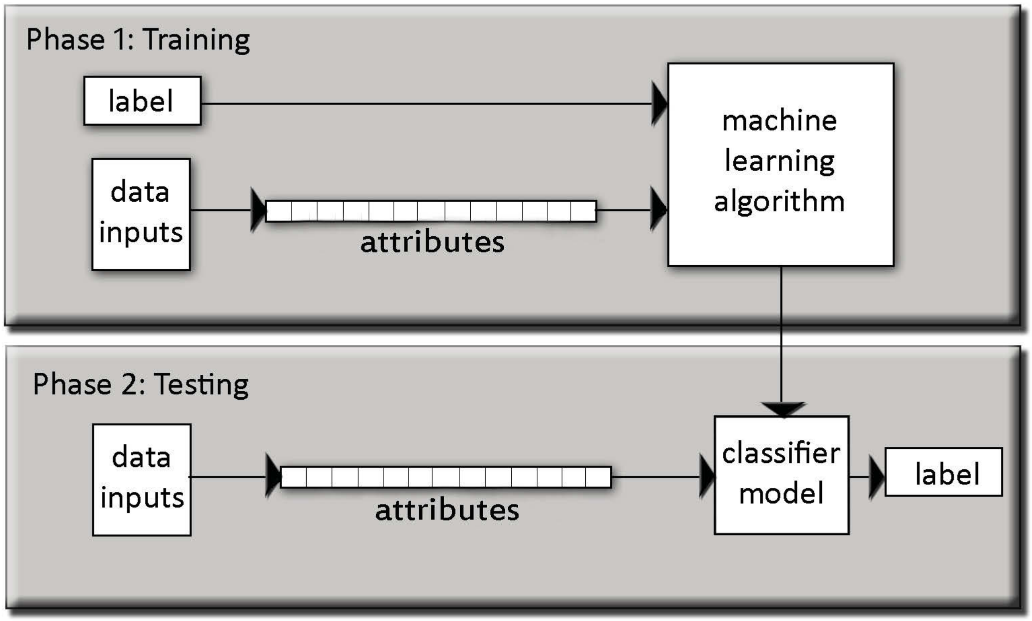


Figure 2.3: The Supervised Learning Process.

selected for this research due to their success in previous traffic classification research [33, 37, 57].

Naïve Bayes. The Naïve Bayes algorithm is a statistical classifier based on the Bayesian theorem [20]. This algorithm analyzes the relationship between the attributes and class label for each data sample in a given dataset to derive a conditional probability for the association between attribute value and class [57]. Naïve Bayes classifiers estimate the probabilities of an attribute having a certain value to predict class membership for each data sample. The classifiers assume that the effect of an attribute value on a given class is independent of the values of the other attributes [20].

Moore *et al.* [33], performed Internet traffic classification using Naïve Bayes. The research distinguished between 10 classes of Internet traffic using six discriminating attributes derived from packet header information. The 10 network traffic classes which included multiple applications per class were labeled: bulk, database, interactive, mail, services, www, P2P, attack, games, and multimedia. The 6 discriminators used to derive attributes for the ML algorithms were: flow duration, TCP port, packet inter-arrival time statistics, payload size statistics, effective bandwidth, and Fourier Transform of the packet inter-arrival time. They achieved a 65% accuracy on per-flow classification when using a simple Naïve Bayes estimator; however, using two refinement techniques on the algorithm, they achieved 95% accuracy [33].

J4.8 decision tree. The J4.8 algorithm is an open source Java implementation of the C4.5 algorithm found in the Weka ML toolkit [60]. Decision tree algorithms use a divide-and-conquer method to classify samples [60]. Figure 2.4 provides an example of a basic decision tree for classifying a fruit with the three structures found in decision trees–nodes, branches and leaves. The nodes in the tree are color, size, and taste, which represent the attributes of a class of fruit with the branches of the tree (e.g., red, green and small) representing the possible values of each attribute. A leaf represents a class label and terminates a series of nodes and branches (i.e., the decisions made by the algorithm) [57]. Determining the class of a sample within a given dataset is a matter of tracing the path of nodes and branches to the terminating leaf [57]. The path for determining a lime class in the decision tree depicted by Figure 2.4 is: if and only if its color is green, its size is small and its taste is sour. Different decision tree algorithms use varying techniques during the training phase to create the final decision tree model used for classification [43].

Barto [3] compared the following five ML algorithms for accuracy when classifying 16 distinct web services within encrypted Transport Layer Protocol (TLS) dataflows: Naïve Bayes, NBTree, LibSVM, J4.8 and AdaBoost+J4.8. The number of packets necessary to accurately identify a web service class is also analyzed. In the research, J4.8 and AdaBoost+J4.8 produced the highest accuracies and runtimes. J4.8 reached a peak accuracy of 97.99% using the first 14 packets of a dataflow. AdaBoost+J4.8 demonstrated a peak accuracy of 98.41% when the first 18 packets of a dataflow are used. The accuracy

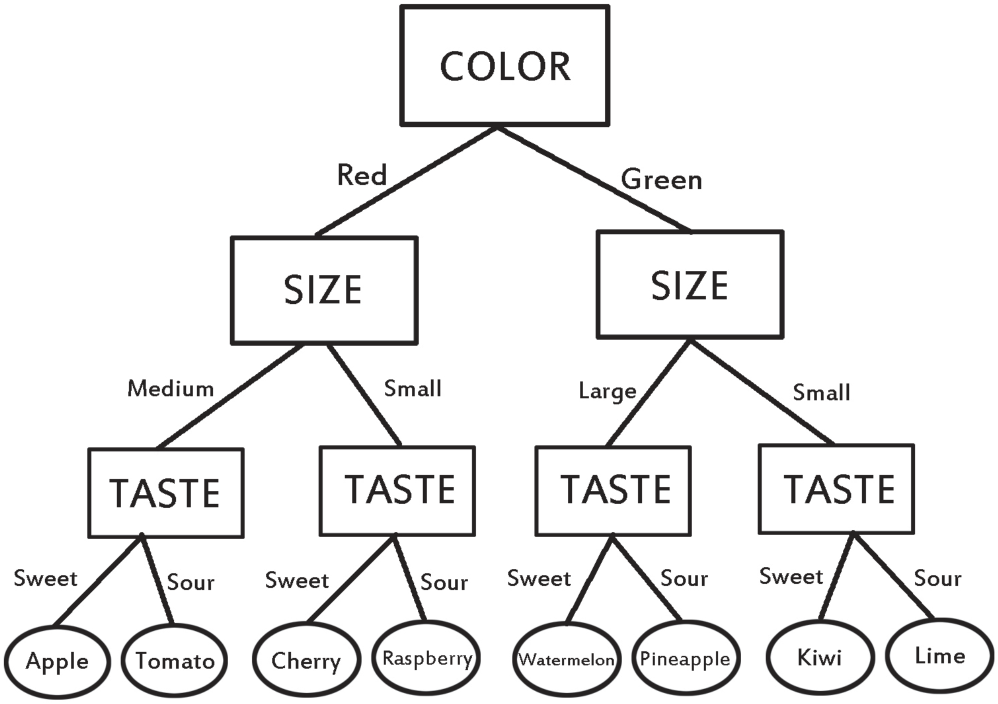


Figure 2.4: Sample Decision Tree.

and runtime results demonstrated the suitability of J4.8 and AdaBoost+J4.8 for real-world detection devices.

Naïve Bayes Tree (NBTree). The NBTree algorithm is a hybrid of a Naïve Bayes classifier and a decision tree classifier, designed to accurately classify increasingly large datasets efficiently [57]. The classification model created during the training phase is described as a decision tree consisting of nodes and branches with Naïve Bayes classifiers on the leaves [20]. Given a node representing an attribute, the algorithm evaluates the utility of a split to create a branch from that node [57]. If there are no splits that provide a significantly better utility, a Naïve Bayes classifier is created and the current node becomes a terminating leaf. NBTree is found to have higher classification accuracy than C4.5 or Naïve Bayes for a majority of the tested datasets [57].

Williams *et al.* [57] evaluate the accuracy and computational performance speed of five supervised ML algorithms, namely, Naïve Bayes (discretisation and kernel density estimation), C4.5 Decision Tree, BayesNet, and NBTree for the classification of six application classes found on a traditional ICT network trace. The six application classes they chose were: FTP-Data, Telnet, SMTP, DNS, HTTP, and Half-Life. They found that the classification accuracy was similar for all five algorithms of at least 95% for six application traffic flows. Nguyen *et al.* [37] found similar results when comparing NBTree with four other supervised ML algorithms for the classification of various IP applications: ftp control, smtp, pop3, imap, https, http and ssh. While all five algorithms demonstrated a high accuracy of up to 99%, NBTree's computational performance speed was significantly slower than the other algorithms when using the full attribute list.

Bayesian Network (BayesNet). The BayesNet algorithm is structured as a combination of a directed acyclic graph of nodes and links, and a set of conditional probability tables [5]. Nodes represent the sample attributes, while links between nodes represent the relationship between the attributes. Conditional probability tables determine the strength of each link [57]. Each node (attribute) has one probability table that defines the probability distribution for each attribute given its parent nodes. If a node has no parents, the probability distribution is unconditional. If it has one or more parent, the probability distribution depends on the values of the parents. BayesNet algorithms use a two-step process to perform classification. The first stage is the learning phase, where local score metrics based on the Bayesian theorem form the initial network structure. The second stage uses an estimation algorithm to create the conditional probability tables. The estimator uses the dataset to calculate class membership probabilities for each sample, as well as the conditional probabilities of each node given its parent nodes in the network structure created in the first stage [57].

Williams *et al.* [57] evaluated BayesNet along with four other supervised ML algorithms, namely Naïve Bayes (discretisation and kernel density estimation), C4.5 Decision Tree, and NBTree for their ability to classify 10 different network traffic classes. They found that given the same features and flow trace, BayesNet provided similar results compared to the other four algorithms all demonstrating at least 95% classification accuracy.

2.4.3 Attribute Reduction Functions.

Previous research has demonstrated the success of using attribute reduction functions for finding optimal attribute subsets while minimizing the loss of classification accuracy [37, 57]. Supervised ML algorithms are provided with a dataset containing data samples. Each sample consists of a list of attribute values and a class assignment used by the training model to gain information about each class and build the classification model. This research uses 24 flow-based attributes calculated from each dataflow to classify SCADA and ICT device dataflows. Previous research has shown that when training a ML classifier, using all available attributes is not always the optimal option because irrelevant or redundant features can negatively impact the algorithm's accuracy and runtime performance [57]. Finding an optimal subset of attributes that maintain or increase the classification accuracy and improve classification time is useful for implementation in real-world traffic classification devices.

The Weka ML toolkit is an open-source data mining application that provides a collection of machine learning algorithms and tools for data preprocessing, classification, regression, clustering, association and visualization [19]. Weka provides two attribute reduction functions that use different methods to find an optimal subset of attributes with minimal loss of classification accuracy. The two attribute reduction functions are the filter function and the wrapper function.

The filter function uses an attribute evaluator and a ranking algorithm to evaluate and rank each attribute in the full attribute set [45]. For example, in this research there are 24 flow-based attributes in the full attribute set; therefore, the filter function assigns each attribute a rank from 1 to 24 based on effectiveness when performing classification. The filter function does not consider the algorithm when ranking the attributes. As such, the same optimal attribute subset is obtained for all four algorithms.

The wrapper function also uses an attribute evaluator to evaluate the performance of all possible attribute subsets using a specified ML algorithm. The wrapper function produces an optimal attribute subset tailored to each algorithm [57]. As such, the wrapper function is run with all four algorithms to obtain an optimal attribute subset unique to each algorithm.

2.4.4 Previous ML Traffic Classification Research.

Previous ML traffic classification research demonstrates the ability to observe distinctive dataflow attributes such as flow duration, byte profiling, packet inter-arrival time, packet size and the distribution of such attributes for a number of TCP applications [10, 12, 26, 27, 40]. Williams *et al.* [57] evaluated five supervised ML algorithms for both accuracy and computational performance: Naïve Bayes (NBD, NBK), C4.5, Bayesian Network and Naïve Bayes Tree. They used publicly available network traces containing millions of dataflows from a variety of applications [57]. They initially defined 22 payload-independent flow attributes to use for training the algorithms; however, using attribute set reduction methods, they found an optimal subset of 9 flow attributes with minimal loss of classification accuracy [57].

Using 10-fold cross-validation to create testing and training sets, they found that with the exception of one algorithm (i.e., Naïve Bayes kernel density estimation) all of the ML algorithms had similar levels of accuracy [57]. The full 22 attribute set had an average of 94.13% classification accuracy and the 9 attribute subset had an average of 93.14%

classification accuracy; however, the computational performance (i.e., training and model build time) improved significantly in 4 out of the 5 algorithms, when using the reduced attribute subset. C4.5 decision tree was the only algorithm that showed no performance speed improvement when using the reduced subset; however, its performance speed was faster than the other four algorithms with both attribute sets [57].

They concluded that when training a ML classifier, using the maximum number of attributes is not always the optimal option and may negatively impact the performance of the algorithm [57]. They also found that the C4.5 ML algorithm had the highest classification accuracy rate and computationally outperformed the other four algorithms with the least training and model build time, even when using the full 22 attribute set. The other four algorithms were all Naïve Bayes-based and, except for the kernel density estimation, had similar classification accuracy and computational performance results with the 22 attribute set and 9 attribute subset, although some had a greater performance increase than others when using the reduced subset.

Li *et al.* [29] used support vector machines (SVM) ML algorithms to train seven classes of applications. They used traffic traces obtained from a campus network backbone and developed a discriminator selection algorithm to find an optimal attribute set for the training phase. Using the optimal attribute subset, their SVM classifier obtained 96.9% accuracy for un-biased (i.e., uniform prior probability) training and testing samples and 99.4% accuracy for biased (i.e., non-uniform prior probability) [29]. Their optimal attribute subset contained 9 attributes, all of which are achievable in real time from captured packet headers, making ML algorithm implementation feasible for real time traffic classification.

Erman *et al.* [13] used the unsupervised ML approach, namely clustering, to demonstrate how cluster analysis can be used to effectively identify groups of Internet traffic that are similar, using only transport layer statistics. They perform experiments using two unsupervised clustering algorithms, K-Means and DBSCAN, to perform network traffic classification [13]. The traces used as input samples to their algorithms come from a publicly available trace from the University of Auckland and a trace collected from the University of Calgary's Internet connection. The results show that both K-Means and DBSCAN work well for Internet traffic classification. K-Means performed with 85% accuracy, while DBSCAN had a 75% classification accuracy rate. Although DBSCAN has a lower overall accuracy, the clusters it forms are more accurate than K-Means, allowing the identification of a significant portion of the trace's connections.

The classification and clustering ML approaches have been the focus of many Internet traffic classification research projects, using only transport layer statistics; however, very little SCADA traffic classification research has been accomplished. Indeed, SCADA network traffic is not generally a class or group that previous research attempts to identify. Therefore, the focus of this research is to use supervised ML algorithms to identify SCADA dataflows within a mixed network traffic trace, independent of packet payload data and to find the optimal subset of flow-based attributes to represent typical SCADA network traffic behavior.

2.5 Summary

This chapter presents background information to understand the research and contributions described in the following chapters. The need for SCADA network traffic behavioral characterization is provided, followed by the current approaches to network analysis and traffic classification. ML algorithms are also discussed with a focus on Naïve Bayes and decision tree supervised learning algorithms, due to their prevalence in this research.

III. Methodology

This chapter describes the methodology for testing four supervised ML algorithms on their ability to identify SCADA network traffic within a mixed network traffic trace as well as identifying an optimal subset of attributes for SCADA network traffic identification.

3.1 Problem Definition

3.1.1 Goals and Hypothesis.

The two main goals of this research are to demonstrate that SCADA network traffic can be accurately identified within a mixed network traffic trace and to identify an optimal subset of attributes with minimal loss of classification accuracy. An algorithm is considered effective if it obtains a TPR of at least .99 for SCADA network traffic and an FPR of $< .05$, the misclassification rate of ICT dataflows as SCADA. The TPR and FPR are both used as measures of effectiveness because it is important to accurately identify SCADA dataflows while at the same time not misclassifying ICT dataflows as SCADA. Supervised ML algorithms have successfully demonstrated accuracies greater than 99% when classifying Internet application traffic [3, 37]; therefore, achieving a TPR of at least .99 for SCADA network traffic identification within a mixed network traffic trace is desirable.

This research explores an optimal subset of attributes from the full 24 flow-based attribute set while maintaining the desired .99 TPR. Utilizing two attribute reduction functions built into the Weka ML toolkit, the filter and wrapper functions, the accuracy of each algorithm is determined for each attribute reduction function subset.

The amount of time each algorithm takes to build its classification model during the training phase and to classify the dataset during the testing phase are also evaluated. While build and classification timing are not the focus of this work, it is beneficial to note them as performance metrics for future work. Indeed, faster classification model build time and dataset classification time will be crucial when implementing ML algorithms in near-real-time network traffic classification tools.

It is expected that given proper training on the unique attributes of SCADA traffic, a supervised ML algorithm will successfully identify SCADA network traffic, obtaining a TPR of at least .99. This classification accuracy provides the ability to identify critical devices communicating on mixed traffic networks. It is expected that an optimal subset of attributes found using the wrapper and filter functions in the Weka ML toolkit will yield a higher TPR lower model build and dataset classification times for all four ML algorithms tested.

3.1.2 Approach.

The first strategic goal of this research is to demonstrate that SCADA network traffic can be identified within a mixed traffic trace, obtaining at least a .99 TPR, using 24 flow-based attributes calculated from each dataflow. The first goal provides the ability to successfully identify SCADA devices communicating on mixed traffic network containing both SCADA and traditional ICT network traffic. The second strategic goal is to identify an optimal subset of attributes, while maintaining the .99 TPR for SCADA network traffic identification. By identifying an optimal subset of attributes that successfully distinguish SCADA dataflows from traditional ICT dataflows, SCADA traffic characteristics can be determined within a mixed traffic network. In addition, measuring the effectiveness of the ML algorithms for accurately classifying each dataflow as either SCADA or ICT increases the ability to identify SCADA devices communicating over a mixed traffic network.

The experiments in this research utilize real-world network traces collected from two different SCADA networks and one traditional ICT network to determine the unique flow-based attributes of each class (i.e., SCADA or ICT). The 24 flow-based attributes calculated from each SCADA and traditional ICT dataflow are used during the training phase to build the ML algorithm's classification model. The model is then used by the ML algorithm's classifier component during the testing phase to classify new dataflows. The ML algorithm's classifier component is provided a dataset containing known, but unlabeled SCADA and ICT dataflows; therefore, the algorithm's effectiveness can be directly measured using the TPR and FPR.

3.2 System Boundaries

The system under test (SUT) is the Dataflow Classification System (DCS) shown in Figure 3.1. There are two components in the DCS. The first component is the ML framework which consists of the Weka ML toolkit version 3.6.10, an open-source data mining application that contains machine learning algorithms and tools for data pre-processing, classification, regression, clustering, association and visualization [19]. The second component is the host computer, a Dell Precision Workstation T7500 with 23.5 GB of memory and a 2.00 GHz x 8 processor. The host computer is running the Ubuntu 13.04 operating system. The two components of the ML framework are the ML trainer, which creates the algorithm's classification model, and the ML classifier, which classifies unlabeled dataflows as either SCADA or ICT using the classification model created by the trainer.

The components under test (CUT) are the ML algorithm's trainer and classifier. Since the ML algorithm's classifier is dependent upon the classification model built by the ML trainer component, both components are tested. During the training phase, hand-classified (labeled) real-world ICT and SCADA dataflows are provided to the SUT's trainer component in order to generate the classification model used in the testing phase. During the testing phase, the SUT's classifier component is provided a dataset with unclassified dataflows and the classification model. It uses the classification model to label each dataflow as belonging to either the SCADA class or the ICT class.

The mixed traffic dataflows are provided to each component as an Attribute-Relation File Format (ARFF) file. The ARFF file format stores each dataflow as a list of attributes and a class label, one dataflow instance per line. When the ARFF file is used for the training phase, each dataflow instance contains the list of attribute values and a class label. However, when the ARFF file is used for testing, each dataflow instance only contains the list of attribute values with no class label.

There are three categories of network traffic behavior that this research hypothesizes will differentiate SCADA and ICT network traffic: packet timing, packet size, and data throughput. The 24 flow-based attributes calculated from each dataflow are categorized into one of the three traffic behavior categories. The class labeled and unlabeled ARFF files are the workload for the SUT.

The SCADA and ICT dataflows used to train and test the DCS are collected from real-world networks. Obtaining real-world SCADA traffic is challenging due to the sensitivity of these critical systems; therefore, only two unique SCADA network traces have been obtained to use with the DCS. Although the operating characteristics may vary among different SCADA networks, the research demonstrates the ability to accurately identify SCADA network traffic within mixed traffic networks utilizing dataflows from two disparate SCADA networks.

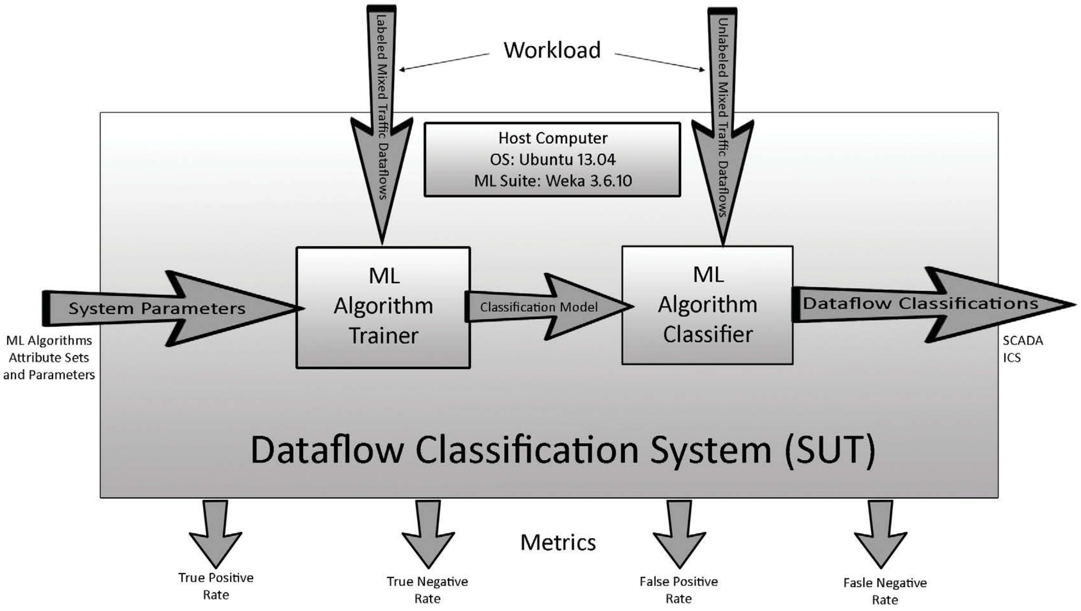


Figure 3.1: Dataflow Classification System.

3.3 Workload

The workload for the DCS consists of the labeled SCADA and ICT device dataflows provided to the ML trainer and the unlabeled dataflows provided to the ML classifier. The mixed traffic datasets used in this research are created using traffic collected from two real-world SCADA networks and one real-world ICT network to ensure accurate classification of each dataflow type.

3.3.1 Network Traffic Collection.

The two SCADA network traces collected for this research are from a real-world water and waste water treatment facility network and an oil and gas company network. The traditional ICT trace was collected from a real-world research network at the Air Force Institute of Technology. Supervised ML algorithms require reasonably sized datasets when performing classification [59]. Therefore, the water and waste water treatment facility trace and the traditional ICT network trace are captured in 1-hour increments over a 24-hour period during a regular business day (i.e., Monday-Friday), creating 24 consecutive 1-hour traces for each network. The oil and gas company network trace is small enough to be captured as one 24-hour trace.

This research makes two assumptions about the collected traces. The first assumption is that each 24-hour trace captures the network activity of a typical day for the SCADA and traditional ICT networks. The second assumption is that the SCADA and ICT networks represent typical networks of their respective type (i.e., the ICT and SCADA networks used in this research possess network traffic behaviors consistent with their respective network types). The facilities that allowed the SCADA network traffic collection will remain anonymous due to the sensitivity of their network information.

3.3.2 Data Preprocessing.

3.3.2.1 Dataflow Classification.

When performing network traffic classification using supervised ML algorithms, a necessary step in dataset preprocessing is to hand-classify each dataflow provided to the training component. In this research, each dataflow must be hand-classified as either SCADA or ICT depending on the device from which the flow originates.

The following network management protocols are present in both the SCADA and traditional ICT network traffic traces: Simple Network Management Protocol (SNMP), Internet Control Message Protocol (ICMP), and Address Resolution Protocol (ARP). The focus of this research is to identify SCADA network traffic from a mixed traffic network trace; therefore, the network management protocols are removed from the traces before using them in the final mixed traffic dataset. All protocols except the Modbus protocol are filtered out of the SCADA network traces, since this is the SCADA protocol used by the two networks. This ensures that only the SCADA dataflows are hand-classified and no other protocols are misclassified. Similarly, all protocols except the TCP protocol are filtered out of the ICT network traffic trace. TCP is used to transport most application traffic such as Lightweight Directory Access Protocol (LDAP), HTTP, HyperText Transfer Protocol Secure (HTTPS), POP3, SMTP, and Modbus. Consequently, it is verified that no Modbus dataflows are present in the ICT traces and only ICT protocols (i.e., non-network management protocols) are hand-classified as ICT dataflows.

3.3.2.2 Mixed Traffic Dataset Creation.

Although the traffic traces from the traditional ICT network and water and waste water facility network were collected in one hour increments many of the files are significant in size. For example, the 1-hour ICT network trace between 1400 to 1500 hours contains 104,121 TCP dataflows and has a file size of 106.3 GB. Due to the requirement for reasonably sized datasets in the ML experiments, only 4 of the 1-hour traces are used in the final dataset. The 4 traces were selected by analyzing characteristics for each of the network's 24 1-hour traces.

The 24 1-hour SCADA files each contained between 400 - 500 Modbus dataflows and the file sizes were all approximately 500 MB regardless of the time of day collected. Given the consistent traffic activity and file size among the 24 1-hour files, four files are selected in increments of approximately 6-hours apart for use in the final mixed traffic dataset. Evenly incremented files are selected in order to capture any variation over the 24-hour period.

The 24 1-hour ICT files revealed that the network traffic was strongly influenced by human activity. Between the hours of 1700 - 0800, the 1-hour traces contained between 15,000 to 30,000 TCP dataflows with file sizes between 1 GB to 2.5 GB. The number of TCP dataflows and file sizes of the 1-hour traces increased significantly during the typical business hours of 0800 to 1700. During that time, the number of TCP dataflows ranged between 32,000 to 108,000 and the file sizes between 2.5 GB to 106.3 GB. Given the inconsistency in number of dataflows and file size, four 1-hour traces are chosen which capture the various traffic activity occurring over the 24-hour period. The number of TCP dataflows found in the four selected ICT traces are: 43,742; 12,763; 26,599 and 31,687, capturing the various activity patterns over the 24-hour period while maintaining a manageable size for the final mixed traffic dataset.

The final mixed traffic dataset is created by combining the dataflows from the four 1-hour water and waste water treatment facility network traces, the 24-hour trace from the oil and gas company network, and the four 1-hour ICT network traces. The final mixed traffic dataset used in this research contains 1,736 SCADA device dataflows and 114,791 ICT device dataflows for a total of 116,527 mixed traffic dataflows. Table 3.1 shows an overview of the trace names and number of dataflows used in this research.

Table 3.1: Overview of the Datasets.

Description	**Number of Dataflows**
Oil and Gas Company Network (24 hours)	3
Water & Waste Water Treatment Facility (1 hour: 2200-2300)	405
Water & Waste Water Treatment Facility (1 hour: 0500-0600)	488
Waste Water Treatment Facility (1 hour: 1000-1100)	448
Water & Waste Water Treatment Facility (1 hour: 2000-2100)	392
Traditional ICT Network (1 hour: 1500-1600)	43742
Traditional ICT Network (1 hour: 0300-0400)	12763
Traditional ICT Network (1 hour: 1000-1100)	26599
Traditional ICT Network (1 hour: 1200-1300)	31687
Final Mixed Traffic Dataset	**116527**

3.3.2.3 Network Traffic Attributes.

The final step in dataset preprocessing is to create the ARFF file provided to the ML training and testing components. An ARFF file consists of three sections: the filename, also referred to as the relation, the attributes, and the data samples. Each of the 116,527 mixed traffic dataflows from the final dataset is a data sample in the ARFF file. Each data sample contains a list of the attribute values and the class assignment for each dataflow. To calculate the attribute values for each dataflow, the nine network traces selected to create the final dataset are split into individual dataflow stream files. For example, the 1-hour ICT network trace between 1200 to 1300 hours contains 31,687 dataflows. Once split, there are 31,686 separate .pcap files, each containing only one TCP dataflow stream. Since there is a total of 116,527 dataflows in the nine selected traces, the attribute values are calculated on 116,527 individual .pcap files and added as one line per dataflow to the ARFF file for use in the experiments.

The research selected 24 flow-based attributes for SCADA dataflow classification. SCADA networks have known attributes that are expected to differentiate their traffic behavior from traditional ICT network traffic. Due to their polling nature, most protocols found on SCADA networks typically have the following characteristics [1, 2]:

- **Deterministic**: In SCADA networks, the operation of a device should be predictable. When a device is given an input a deterministic output is expected since the critical processes are dependent on predictability.
- **Hierarchical**: Devices communicate in a one-to-many fashion. In a SCADA network, a master device polls many field devices for operational information.
- **Consistent**: SCADA master devices poll field devices in set time intervals. The periodicity of packets sent between these devices occurs in steady intervals rather than traditional ICT devices, such as email servers, where human interaction affects

sent packet timing. Furthermore, the topology of SCADA networks tends to remain more static than traditional ICT networks because nodes are not added or removed as often.

Three categories of network traffic behavior are selected based on prior knowledge of typical SCADA protocol behavior: packet timing, packet size, and data throughput. The 24 flow-based statistics, which become the 24 attributes for the ML algorithms, are shown in Table 3.2. Attributes 1 to 8 relate to packet timing, attributes 9 to 16 relate to packet size, and attibutes 17 to 24 relate to data throughput. The selected attributes require basic arithmetic functions for calculations, making their speed of calculation desirable for implementation in near-real-time devices. The 24 attributes are calculated from each dataflow stream without the need to determine ports, protocol, IP addresses, or payload content; only flow-based information such as packet inter-arrival time and size are required. In addition to the 24 attributes, each dataflow is given a label based on its class membership. This research uses two labels: SCADA or ICT. The goal of the DCS is to correctly classify unlabeled dataflows from a mixed traffic trace as belonging to either the SCADA or ICT class.

Table 3.2: Full Attribute Set.

Number	**Category**	**Name**	**Description**
1	Timing	min_iat_ab	Minimum inter-packet arrival time (src to dest)
2	Timing	max_iat_ab	Maximum inter-packet arrival time (src to dest)
3	Timing	mean_iat_ab	Mean inter-packet arrival time (src to dest)
4	Timing	var_iat_ab	Variance of inter-packet arrival time (src to dest)
5	Timing	min_iat_ba	Minimum inter-packet arrival time (dest to src)
6	Timing	max_iat_ba	Maximum inter-packet arrival time (dest to src)
7	Timing	mean_iat_ba	Mean inter-packet arrival time (dest to src)
8	Timing	var_iat_ba	Variance of inter-packet arrival time (dest to src)
9	Size	min_efb_ab	Minimum Ethernet frame byte size (src to dest)
10	Size	max_efb_ab	Maximum Ethernet frame byte size (src to dest)
11	Size	mean_efb_ab	Mean Ethernet frame byte size (src to dest)
12	Size	var_efb_ab	Variance of Ethernet frame byte size (src to dest)
13	Size	min_efb_ba	Minimum Ethernet frame byte size (dest to src)
14	Size	max_efb_ba	Maximum Ethernet frame byte size (dest to src)
15	Size	mean_efb_ba	Mean Ethernet frame byte size (dest to src)
16	Size	var_efb_ba	Variance of Ethernet frame byte size (dest to src)
17	Throughput	min_bps_ab	Minimum bits per second (src to dest)
18	Throughput	max_bps_ab	Maximum bits per second (src to dest)
19	Throughput	mean_bps_ab	Mean bits per second (src to dest)
20	Throughput	var_bps_ab	Variance of bits per second (src to dest)
21	Throughput	min_bps_ba	Minimum bits per second (dest to src)
22	Throughput	max_bps_ba	Maximum bits per second (dest to src)
23	Throughput	mean_bps_ba	Mean bits per second (dest to src)
24	Throughput	var_bps_ba	Variance of bits per second (dest to src)

3.4 Performance Metrics

The DCS provides a binary classification service because it only considers two labels or classes: SCADA and ICT. Accuracy is often used to measure the performance of a classifier, and is sometimes the only evaluation criteria given in ML research [30]. The total accuracy of a classifier is defined as the number of correct predictions it makes over the total number of predictions made [23].

Reporting the number of True positives (TPs), True negatives (TNs), False positives (FPs), and False negatives (FNs) and their associated rates provides a representation of a binary classifier's performance [47]. Based on Japkowicz *et al.* [23], the following definitions are used for this research:

- **TP** = The number of SCADA dataflow instances correctly classified as SCADA.
- **FP** = The number of ICT dataflow instances incorrectly classified as SCADA.
- **TN** = The number of ICT dataflow instances correctly classified as ICT.
- **FN** = The number of SCADA dataflow instances incorrectly classified as ICT.

The first goal is to demonstrate a TPR of at least .99 for identifying SCADA network traffic, therefore the effectiveness of each ML algorithm is measured using the number of TPs and FPs and their associated rates. The TPR is calculated using Equation 3.1 and the FPR is calculated using Equation 3.2. A ML algorithm is considered effective if it achieves a TPR of at least .99 and a FPR of $< .05$. The TPR and FPR are both used as measures of effectiveness because it is important to accurately identify SCADA dataflows while at the same time minimizing the misclassification of ICT dataflows as SCADA. Note that the false negative rate (FNR) is 1 - TPR and, as such, is implicitly included in the TPR results. The true negative rate (TNR) can be found using the equation TNR = 1 - FPR.

$$TPR = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.1}$$

$$FPR = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{3.2}$$

The time each algorithm takes to build the classification model and to classify the dataset are also used as performance metrics for each experiment to aid in deciding which algorithm to implement in a near-real-time classification device.

3.5 Factors

Two factors are selected for testing and evaluation in this research as shown in Table 3.3.

Table 3.3: Factors and Levels.

Factors	Levels
ML Algorithm	Naïve Bayes, NBTree, BayesNet, J4.8 Decision Tree
Attribute Set	Full Attribute Set, Wrapper Attribute Subset, Filter Attribute Subset

3.5.1 ML Algorithms.

Four ML algorithms are chosen for the experiments: Naïve Bayes, NBTree, BayesNet, and J4.8 Decision Tree. All four are supervised-learning algorithms; therefore, they take training datasets with labeled dataflows from both SCADA and ICT network traffic and create a classification model based on the given dataset. The ML classifier component uses the classification model created by the trainer to classify the instances of a given unlabeled dataset. The model makes a determination based on information gained during the training phase as to what class each dataflow belongs.

3.5.2 Attribute Sets.

There are three attribute set levels tested with each ML algorithm in this research. The Weka ML toolkit provides two attribute reduction functions (i.e., wrapper and filter) to find an optimal subset of attributes that yield a minimal loss of classification accuracy. This research evaluates the full attribute set and both optimal attribute subsets produced by the wrapper and filter functions.

Full Attribute Set. Twenty-four flow-based statistics are selected as the dataflow attributes for this research. Each of the 24 attributes relate to one of three traffic behavior categories: packet timing, packet size, and data throughput. The three categories are based on the known characteristics of SCADA protocols as deterministic, hierarchical, and consistent due to their polling nature [1, 2]. The 24 attributes make up the full attribute set for this research.

Wrapper Attribute Subset. The wrapper function in Weka utilizes the attribute evaluator "ClassifierSubsetEval" which creates all possible subsets from the full attribute set [44]. A classification algorithm is specified by the experimenter which allows the function to find an optimal subset of attributes tailored to a specific algorithm [44]. The wrapper function provides an optimal subset of attributes that minimize the loss of classification accuracy in comparison to the full attribute list's TPR [44]. The wrapper function is run with each of the four ML algorithms to reveal the optimal attribute subset.

Filter Attribute Subset. The filter function relies on the attribute evaluator "InfoGainAttributeEval" and a ranking algorithm to evaluate and rank all 24 attributes in the full attribute set [45]. Each attribute is assigned a rank from 1 to 24 based on its effectiveness when performing classification. Attribute rank is assigned by the filter function's evaluator and ranking algorithm; the classification algorithm is not considered. As a result, the attribute rank applies to all four ML algorithms.

Overfitting is possible when lower-ranked attributes are eliminated prior to classification. Overfitting in ML occurs when the trainer component creates a classification model that is too general to accurately classify new data samples [60]. The top five ranked attributes are selected as the filter attribute subset and the lower 19 ranked attributes are eliminated during classification. To compensate for the possibility of overfitting, 10-fold cross-validation is used to take advantage of all available data samples for training and testing [49] and multiple experiment repetitions.

3.6 System Parameters

The DCS parameters include the selected ML algorithms and the individual algorithm parameters specified within Weka. The host computer specifications can affect the amount of time the ML algorithm's trainer and classifier components take to build the classification model and classify the dataset. Note that the focus of this research is the classification accuracy of the SUT. As such, the host computer specifications are not a parameter for this research.

The research tests four supervised ML algorithms: Naïve Bayes, NBTree, BayesNet, and J4.8 Decision Tree. The default parameters are selected in Weka for the four algorithms consistent with previous research [28, 33, 37, 57] and pilot experiment results.

3.7 Evaluation Technique

The evaluation technique is direct measurement of the DCS classification accuracy for each algorithm and attribute set tested. Direct measurement on the DCS is used since the network traffic and dataflow attributes are collected from real-world SCADA and ICT networks.

The test equipment for the experiments consists of an Ubuntu 13.04 computer, the Weka ML toolkit (version 3.6.10), and the network traffic analysis tool Wireshark (version 1.8.2).

The research uses TPR (Equation (3.1)) and FPR (Equation (3.2)) as the primary means of evaluating and comparing learning algorithms. Supervised ML algorithms have successfully demonstrated accuracies greater than 99% when classifying Internet application traffic [37]; therefore, a TPR of $> .99$ is considered successful for the primary research goal. Additionally, an FPR of $< .05$ is considered effective because it is also important to not misclassify ICT dataflows as SCADA.

Classification model build time and dataset classification time are also reported for each algorithm and attribute set combination in order to provide a means for evaluating the algorithms for possible implementation into a real-world traffic classification device. Network traffic classification should occur in near-real-time; therefore, the faster the build and classification time of an algorithm with a given attribute set, the more feasible for use on a real network.

3.7.1 Cross-Validation.

Performing *k*-fold cross-validation takes full advantage of all available data samples for training and testing [49]. When performing *k*-fold cross-validation using the Weka ML toolkit, the dataset instances are randomized and stratified, meaning each fold contains approximately the same percentage of labels as the overall dataset [46]. Stratification yields a less biased estimate of true accuracy [25]. The dataset is then split into *k* equally-sized folds, each fold being a subset of the original dataset [47]. A total of *k* rounds of learning are performed, each round utilizing a different fold of the *k* folds as the testing dataset and the remaining folds for the training dataset [47]. The final averaged accuracy from the *k* rounds provides a true estimate of the accuracy of the learning algorithm on the given dataset [49].

3.7.2 10-Fold Cross-Validation.

Testing a classification model with data samples that were used during the training phase can lead to higher classification accuracy rates than the algorithm would typically yield [59]. ML research is generally more concerned with how a classifier performs with data samples it has never seen before, rather than how well it can repeat the classification labels on the data samples used in training. To avoid this dilemma, a 10-fold cross-validation is used to measure the classification accuracy of the four ML algorithms. Note that common values when performing cross-validation are five or ten [49]. It has been shown that $k = 10$ provides a true estimate of an algorithm's classification performance [59].

Figure 3.2 illustrates the first three rounds of a 10-fold cross-validation experiment using a dataset containing two classes. In this figure the oval represents one class (i.e., ICT) and the diamond represents the other class (i.e., SCADA). The dataset containing all data samples is divided into ten equally-sized folds. Each fold contains an equal number of data samples and is stratified to ensure an equal ratio of class type for each fold. For example, in this research there are 116,527 total data samples in the final dataset, of which 1,736 are classified as SCADA and 114,791 are classified as ICT. Therefore, when 10-fold cross-validation is performed on the final dataset, each fold contains approximately 11,652 data samples, of which approximately 173 are classified as SCADA and 11,479 as ICT.

When performing 10-fold cross-validation, 9 folds are used for training and 1 fold is used for testing. Using more data samples for training allows the trainer to create a more accurate classification model [59]. When a fold is used as the testing dataset, the data sample's class label attribute is removed prior to providing the dataset to the classifier component. Round one in Figure 3.2 shows fold one used as the testing dataset and folds two through ten used as the training dataset. Round two shows fold two as the testing dataset and the other nine folds used for the training dataset. Round three shows fold three as the testing dataset and the other nine folds used for training. Ten rounds are conducted to ensure each fold is used as a testing dataset; therefore, each data sample has been classified at least once by the classifier component. The average classification accuracy after all 10 rounds provides the final accuracy of the selected algorithm using the given dataset.

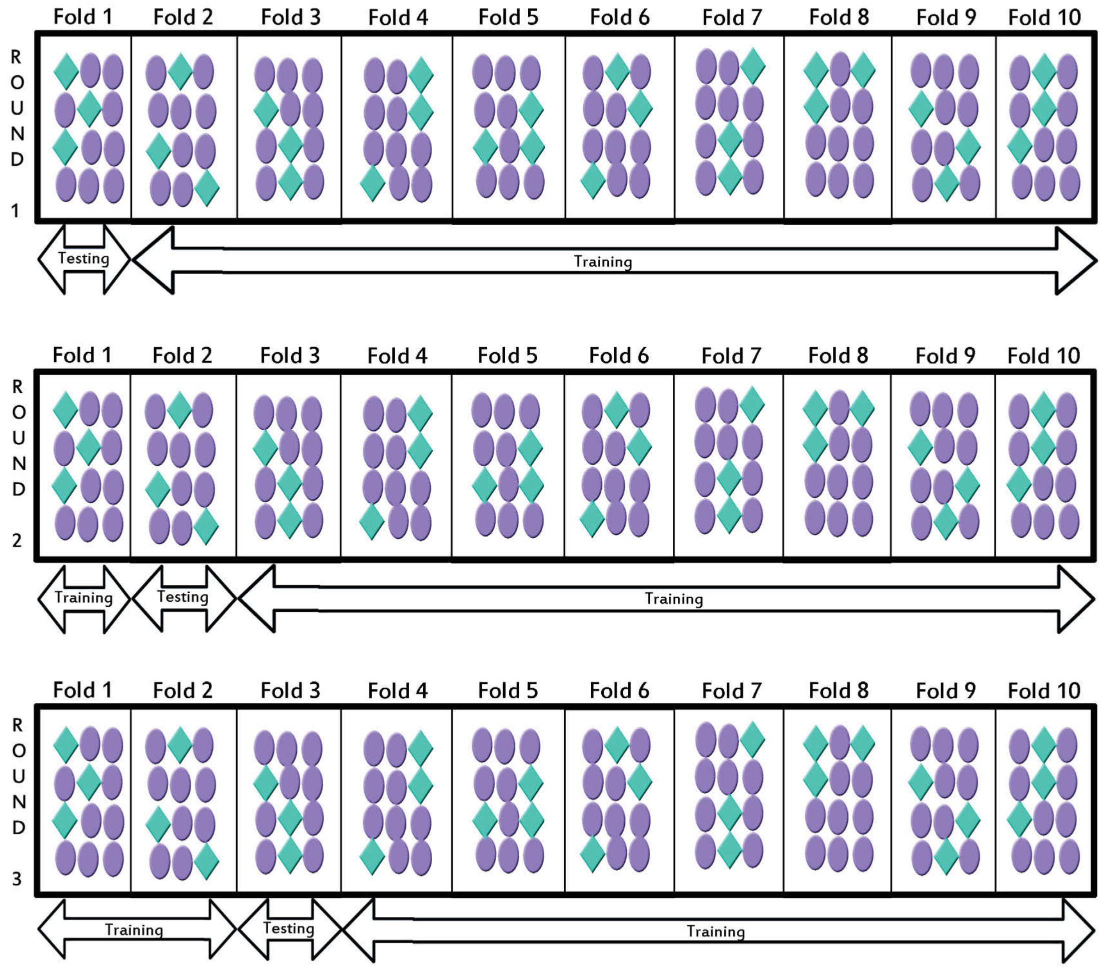


Figure 3.2: Three Iterations of 10-Fold Cross-Validation.

.

3.8 Experimental Design

A partial factorial design for the experiments is used in this research. Experiments are conducted for all four ML algorithms using each of the three attribute levels: the full attibute set, the wrapper attribute subset, and the filter attribute subset. Each attribute set is tested with the four algorithms for 3 * 4 = 12 experiments. Even though 10-fold cross-validation is used for every experiment, ten repetitions of each experiment are run to ensure a true estimate of the algorithm's accuracy performance is achieved. With 12 unique experiments and 10 repetitions, 120 total experiments are conducted. Weka's 10-fold cross-validation fold generator automatically creates random, stratified folds; therefore, the entire dataset can be used for each experiment.

Figure 3.3 is an example of the steps taken in Weka to perform one full experiment using 10-fold cross-validation and the Naïve Bayes ML algorithm. It begins by loading the ARFF file which contains the workload (mixed dataflows), then the Class Assigner specifies which attribute contains the class type (i.e., SCADA or ICT). Next, the labeled dataset goes to the CrossValidation FoldMaker to create the 10 folds. For each round of the 10-fold cross-validation, 9 labeled folds are sent to the training component of the selected supervised ML algorithm and 1 unlabeled fold is sent to the classifier component. This portion of the experiment is conducted 10 times (10 rounds), so that each fold is used for training and testing. The average of the 10 rounds is sent to the Classifier Performance Evaluator, which displays the results in a Text Viewer. Figure 3.4 shows an example output of the TextViewer after one experiment is run using 10-fold cross-validation on a dataset.

The confusion matrix at the bottom of Figure 3.4 provides the number of TPs, TNs, FPs, and FNs. In this example, there are a total of 1,694 TPs; 42 FNs; 114,177 TNs; and 614 FPs. Indeed, 1694 SCADA dataflows are classified accurately and 42 misclassified as ICT for a TPR .976. Similarly, 614 misclassified as SCADA and 114,117 ICT dataflows are classified accurately for an FPR of .005.

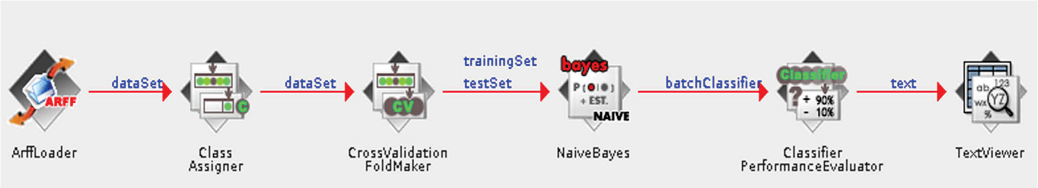
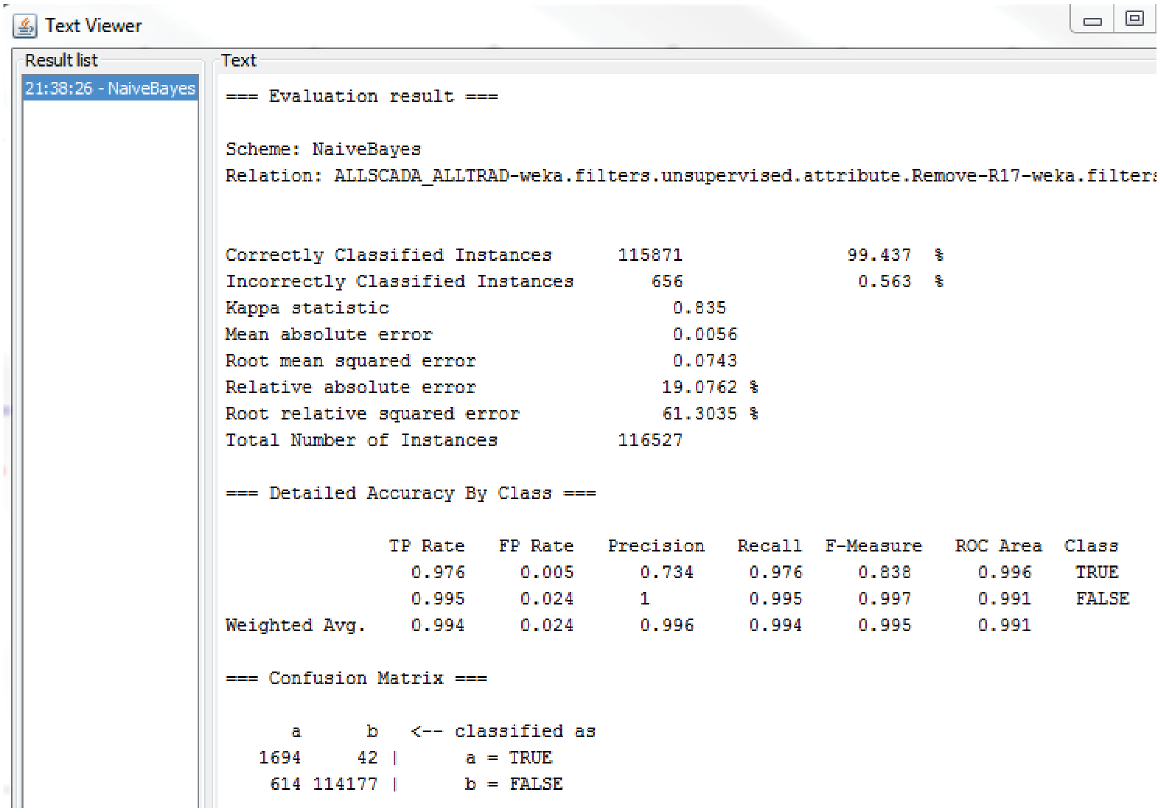


Figure 3.3: Steps to Conduct One ML Experiment in Weka.

.



```
=== Evaluation result ===

Scheme: NaiveBayes
Relation: ALLSCADA_ALLTRAD-weka.filters.unsupervised.attribute.Remove-R17-weka.filters


Correctly Classified Instances      115871               99.437  %
Incorrectly Classified Instances       656                0.563  %
Kappa statistic                          0.835
Mean absolute error                      0.0056
Root mean squared error                  0.0743
Relative absolute error                 19.0762 %
Root relative squared error             61.3035 %
Total Number of Instances           116527

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.976     0.005      0.734     0.976     0.838      0.996    TRUE
                 0.995     0.024      1         0.995     0.997      0.991    FALSE
Weighted Avg.    0.994     0.024      0.996     0.994     0.995      0.991

=== Confusion Matrix ===

      a      b   <-- classified as
   1694     42 |      a = TRUE
    614 114177 |      b = FALSE
```

Figure 3.4: TextViewer Results from One ML Experiment Run in Weka.

.

3.9 Methodology Summary

This chapter examined the methodology used to test four supervised ML algorithms, Naïve Bayes, NBTree, BayesNet, and J4.8 decision tree, for identifying SCADA network traffic in a mixed network traffic trace. A mixed traffic dataset containing both SCADA and ICT device dataflows is created using traffic collected from two real-world SCADA networks and a real-world ICT network. Twenty-four flow-based attributes categorized into one of the three traffic behavior categories: packet timing, packet size, and data throughput, are calculated from each dataflow in the dataset. Two attribute reduction functions are used to find optimal attribute subsets for the algorithms with minimal loss of classification accuracy. The effectiveness of each algorithm is measured using the average TPR and FPR after performing 10 repetitions of 10-fold cross-validation experiments.

IV. Results and Analysis

There are two main goals for this research. The first goal is to demonstrate the ability to identify SCADA network traffic within a mixed network traffic trace by achieving a TPR of at least .99. Section 4.1 presents the results of each algorithm's SCADA network traffic identification when using the full attribute list. The second goal is to identify an optimal subset of attributes while maintaining the .99 TPR for SCADA network traffic. Section 4.2 presents the resulting attribute subsets from the attribute reduction functions in Weka and their TPR and FPR when tested with each algorithm. Section 4.3 reports the time to build the classification model and to classify the dataset for each algorithm and attribute set level for possible implementation.

4.1 Algorithm Accuracy Analysis

The first goal is to demonstrate the ability to identify SCADA network traffic using supervised ML algorithms given a mixed network traffic trace obtaining at least a .99 TPR. Twenty-four flow-based attributes are calculated from each dataflow in the dataset to be used by the ML algorithm for traffic classification. Each of the four ML algorithms is tested with the full attribute set using 10-fold cross-validation with 10 repetitions. The TPR and FPR are used to measure the effectiveness of each ML algorithm for identifying SCADA network traffic. TPR is the rate of accurately identified SCADA dataflows and FPR is the rate of ICT dataflows misclassified as SCADA. TPR is calculated using Equation (3.1) and FPR is calculated using Equation (3.2).

When using 10-fold cross-validation, each round utilizes one fold of the dataset is used for testing and nine folds are used for training. There are a total of 10 rounds per experiment to ensure each fold is used at least once for testing and training. Therefore, the number of testing samples is always 1/10th the number of samples in the full dataset.

The final dataset contains 116,527 mixed network traffic data samples, of which approximately 104,874 are used for testing and 11,653 for training at each round. The folds are stratified to ensure an even ratio of class type per fold. For example, there are 1,736 SCADA dataflows and 114,791 ICT dataflows in the final dataset. On average, each of the folds used in 10-fold cross-validation contains 173 SCADA dataflows and 11,479 ICT dataflows; however, this number varies slightly since Weka generates random folds at each round.

Table 4.1 provides the average TPR and FPR after 10 repetitions of 10-fold cross-validation for each algorithm using the full attribute list. With a TPR of .9933, BayesNet is the only algorithm that demonstrates the ability to classify SCADA dataflows in a mixed traffic network with at least a .99 TPR using the full attribute set. While the other three ML algorithms display TPRs of greater than .96, they do not meet the first research goal. All four algorithms meet the goal of an FPR < .05 for ICT dataflows misclassified as SCADA, with J4.8 having the lowest FPR at .0023.

Table 4.1: Algorithm Accuracies using Full Attribute Set.

Algorithm	**TPR**	**FPR**
Naïve Bayes	.9762	.0052
NBTree	.975	.0026
BayesNet	.9933	.0037
J4.8	.9668	.0023

4.2 Optimal Subsets of SCADA Dataflow Attributes

The second goal of this research is to find an optimal subset of attributes that maintains at least a .99 TPR for SCADA network traffic identification. The Weka ML toolkit contains two attribute reduction functions (i.e., wrapper and filter) that provide an optimal subset of attributes while minimizing loss of classification accuracy.

4.2.1 Wrapper Function Results.

The wrapper function in Weka utilizes the attribute evaluator “ClassifierSubsetEval” which creates all possible subsets from the full attribute set [44]. A classification algorithm is specified by the experimenter which allows the function to find an optimal subset of attributes tailored to the specified algorithm [44]. The Weka wrapper function derives an optimal subset of attributes from the full attribute set by discerning the attributes that minimize the loss of classification accuracy [44]. Note that the wrapper function does not rank the attributes in the given subset. The identified optimal wrapper attribute subsets specific to each algorithm are:

- **Naïve Bayes**:

  13 - Minimum Ethernet frame byte size (source to destination)

  14 - Maximum Ethernet frame byte size (destination to source)

  15 - Mean Ethernet frame byte size (destination to source)

  19 - Mean bits per second (source to destination)

  23 - Mean bits per second (destination to source)

  24 - Variance of bits per second (destination to source)

- **NBTree**:

  1 - Minimum inter-packet arrival time (source to destination)

  5 - Minimum inter-packet arrival time (destination to source)

  6 - Maximum inter-packet arrival time (destination to source)

  7 - Mean inter-packet arrival time (destination to source)

11 - Mean Ethernet frame byte size (source to destination)

13 - Minimum Ethernet frame byte size (source to destination)

16 - Variance of Ethernet frame byte size (destination to source)

19 - Mean bits per second (source to destination)

- **BayesNet**:

  1 - Minimum inter-packet arrival time (source to destination)

  15 - Maximum Ethernet frame byte size (destination to source)

  23 - Mean bits per second (destination to source)

- **J4.8 Decision Tree**:

  1 - Minimum inter-packet arrival time (source to destination)

  2 - Maximum inter-packet arrival time (source to destination)

  3 - Mean inter-packet arrival time (source to destination)

  4 - Variance of inter-packet arrival time (source to destination)

  5 - Minimum inter-packet arrival time (destination to source)

  9 - Minimum Ethernet frame byte size (source to destination)

  13 - Minimum Ethernet frame byte size (source to destination)

  16 - Variance of Ethernet frame byte size (destination to source)

  20 - Variance of bits per second (source to destination)

  23 - Mean bits per second (destination to source)

  24 - Variance of bits per second (destination to source)

The list of wrapper attribute subsets is provided in Table 4.2. Note that the NBTree, BayesNet, and J4.8 Decision Tree subsets include attributes associated with all three traffic behavior categories (i.e., packet timing, packet size and data throughput), whereas the Naïve Bayes subset only uses packet size and data throughput. The attributes selected for each algorithm differ because their unique classification models require certain attributes for classification while other attributes may be unnecessary to obtain the same accuracy rate.

Table 4.3 provides the average TPR and FPR after 10 repetitions of 10-fold cross-validation. Each algorithm is run using their unique optimal subset of attributes provided by the wrapper function found in Table 4.2.

Table 4.2: Wrapper Attribute Subset.

Algorithm	**Wrapper Attribute Subset**
Naïve Bayes	13, 14, 15, 19, 23, 24
NBTree	1, 5, 6, 7, 11, 13, 16, 19
BayesNet	1, 15, 23
J4.8 Decision Tree	1, 2, 3, 4, 5, 9, 13, 16, 20, 23, 24

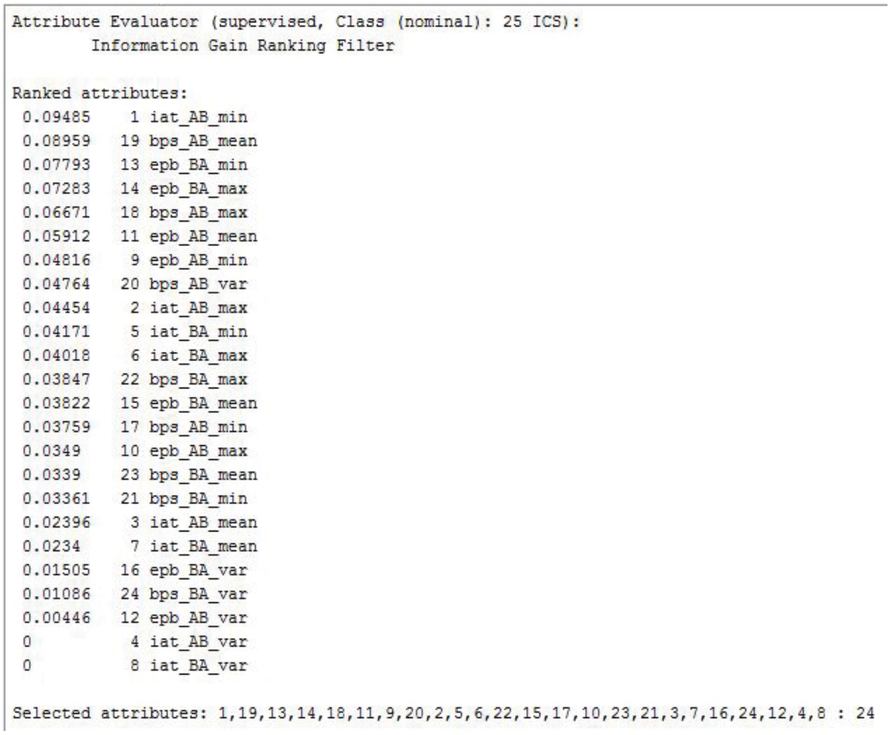
Table 4.3: Algorithm Accuracies using Wrapper Attribute Subset.

Algorithm	**TPR**	**FPR**
Naïve Bayes	.9838	.0041
NBTree	.9805	.0025
BayesNet	.9709	.0029
J4.8	.9632	.0022

The SCADA network traffic identification results for Naïve Bayes and NBTree improve by almost .02 TPR when using the wrapper attribute subset; however, their accuracies do not meet the goal of at least .99. BayesNet and J4.8 show a decrease in their SCADA network traffic identification results when using the wrapper attribute subset. BayesNet demonstrated a TPR of .9933 for SCADA network traffic identification when using the full attribute list and only .9709 with the wrapper attribute subset, no longer meeting the goal of at least .99. All four algorithms show a slight decrease in their FPRs, indicating that the wrapper attribute subsets reduce the erroneous classification of ICT dataflows as SCADA dataflows. While none of the algorithms meet the goal of at least a .99 TPR for SCADA network traffic identification when using the wrapper attribute subset, Naïve Bayes demonstrates the highest TPR of .9838.

4.2.2 Filter Function Results.

The filter function in Weka relies on the attribute evaluator "InfoGainAttributeEval" and a ranking algorithm to evaluate and rank all 24 attributes in the full attribute set [45]. Attribute rank is assigned by the function's evaluator and ranker; therefore, the classification algorithm is not considered. As such, the attribute rank applies to all four ML algorithms. Figure 4.1 shows the output after running the filter attribute reduction function on the mixed traffic dataset. The rank and associated percentage of information gain for each attribute is provided in the output. The fully ranked list of attributes provided by the filter function in order of precedence is: 1, 19, 13, 14, 18, 11, 2, 9, 20, 5, 6, 15, 17, 22, 10, 21, 23, 3, 7, 16, 24, 4, 12, 8.



```
Attribute Evaluator (supervised, Class (nominal): 25 ICS):
        Information Gain Ranking Filter

Ranked attributes:
 0.09485    1 iat_AB_min
 0.08959   19 bps_AB_mean
 0.07793   13 epb_BA_min
 0.07283   14 epb_BA_max
 0.06671   18 bps_AB_max
 0.05912   11 epb_AB_mean
 0.04816    9 epb_AB_min
 0.04764   20 bps_AB_var
 0.04454    2 iat_AB_max
 0.04171    5 iat_BA_min
 0.04018    6 iat_BA_max
 0.03847   22 bps_BA_max
 0.03822   15 epb_BA_mean
 0.03759   17 bps_AB_min
 0.0349    10 epb_AB_max
 0.0339    23 bps_BA_mean
 0.03361   21 bps_BA_min
 0.02396    3 iat_AB_mean
 0.0234     7 iat_BA_mean
 0.01505   16 epb_BA_var
 0.01086   24 bps_BA_var
 0.00446   12 epb_AB_var
 0          4 iat_AB_var
 0          8 iat_BA_var

Selected attributes: 1,19,13,14,18,11,9,20,2,5,6,22,15,17,10,23,21,3,7,16,24,12,4,8 : 24
```

Figure 4.1: Filter Function Attribute Rank.

The top five ranked attributes chosen as the filter attribute subset for the final experiments, shown in order of precedence, are:

- 1 - Minimum inter-packet arrival time from source to destination
- 19 - Mean bits per second from source to destination
- 13 - Minimum Ethernet frame byte size from destination to source
- 14 - Maximum Ethernet frame byte size from destination to source
- 18 - Maximum bits per second from source to destination

Note that the top five ranked attributes include attributes from all three traffic behavior categories–packet timing, packet size, and data throughput. The filter function ranks minimum inter-packet arrival time as the most effective attribute for classification accuracy, which is associated with the packet timing category. The mean bits per second attribute is ranked second and is associated with the data throughput category. The third and fourth ranked attributes are associated with the packet size category and the fifth ranked attribute is associated with data throughput. Indeed, for the given mixed traffic dataset, packet timing is the most influential factor for identifying SCADA network traffic.

Table 4.4 provides the average TPR and FPR after 10 repetitions of 10-fold cross-validation. Each algorithm is run using the top five ranked attributes provided by the filter function.

Table 4.4: Algorithm Accuracies using Filter Attribute Subset.

Algorithm	**TPR**	**FPR**
Naïve Bayes	.976	.0078
NBTree	.968	.0026
BayesNet	.9935	.0051
J4.8	.9706	.0025

Naïve Bayes and NBTree show a slight decrease in their SCADA network traffic identification results when using the filter attribute subset compared to both the wrapper attribute subset and full attribute set. For both algorithms, the wrapper attribute subset is optimal as it provided the highest TPRs; however, neither algorithm met the .99 TPR goal. J4.8 displays a slight improvement when using the filter attribute subset over both the full attribute set (+.0038) and the wrapper attribute subset (+.0074); therefore, the filter attribute subset is its optimal attribute subset, although J4.8 never reached the .99 accuracy goal.

BayesNet shows a peak TPR of .9935 when using the filter attribute subset making it an optimal subset for the algorithm. All algorithms, except NBTree, show a slight FPR increase in comparison to using the full attribute list. This increase indicates that overfitting may have occurred because the filter attribute subset misclassifies more ICT dataflows as SCADA, however, all FPRs remain under .01, which meets the goal of less than .05.

Although Naïve Bayes, NBTree, and J4.8 do not meet the .99 TPR goal, they demonstrate SCADA network traffic identification results greater than .95 with all three attribute set levels. Moore *et al.* [33], achieved a 65% accuracy for Internet traffic classification when using Naïve Bayes and a 95% accuracy after two refinement techniques were applied; therefore, a TPR of at least .95, while not meeting the overall research objective, is still a successful result.

Overall, BayesNet using the filter attribute subset demonstrates the highest SCADA network traffic identification results with a TPR of .9935. Furthermore, BayesNet is the only ML algorithm tested that meets the first goal of identifying SCADA network traffic, obtaining at least a .99 TPR, and the second goal of finding an optimal subset of attributes while maintaining the .99 TPR. BayesNet's optimal attribute subset contains attributes from all three traffic behavior categories. Additionally, the top five ranked attributes from the filter function include attributes from all three traffic behavior categories. This reinforces the distinction of SCADA network traffic from traditional ICT network traffic based on packet timing, packet size, and data throughput.

4.3 Algorithm Timing Comparison

The classification model build time and dataset classification time for each algorithm and attribute set level were also examined to provide additional performance metrics for evaluating the algorithms for possible implementation into a real-world traffic classification device. Table 4.5 provides the mean model build time (MBT) and dataset classification time (DCT) results in seconds. Note that the TPR is included as a reference for consideration when comparing algorithm performance.

Table 4.5: Mean Model Build and Dataset Classification Times.

	Full Attribute Set			**Wrapper Attribute Subset**			**Filter Attribute Subset**		
	MBT (s)	**DCT (s)**	**TPR**	**MBT(s)**	**DCT (s)**	**TPR**	**MBT (s)**	**DCT (s)**	**TPR**
Naïve Bayes	1.327	13.873	.9762	.324	6.076	.9838	.229	7.571	.976
NBTree	339.226	1969.574	.975	37.432	236.668	.9805	20.942	131.458	.968
BayesNet	10.906	68.994	.9933	1.022	12.878	.9709	1.799	17.301	.9935
J4.8	13.161	71.339	.9668	9.799	55.301	.9632	1.97	20.33	.9706

4.3.1 Model Build Time.

Table 4.5 displays the mean classification model build times for each algorithm and attribute set level tested. The fastest model build time is underlined in the table for each attribute set level. Naïve Bayes had the fastest model build times for all three attribute set levels: 1.327 seconds with the full attribute set, .324 seconds with the wrapper attribute subset, and .229 seconds with the filter attribute subset. NBTree had the slowest model build times for all three attribute set levels. Note that BayesNet, the only ML algorithm tested that exceeded the desired .99 TPR for SCADAnetwork traffic identification, had a mean model build time of 10.906 seconds with the full attribute list and 1.799 seconds with the filter attribute subset.

While classification model build time provides a metric for algorithm performance, it is only necessary to build the model once when classifying continuous datasets. Therefore, a more important performance discriminator is the dataset classification time when choosing an algorithm for implementation into a real-world device. Note that the model build times improve significantly for all four algorithms when the wrapper and filter attribute subsets are used in the experiments.

4.3.2 Dataset Classification Time.

Table 4.5 also shows the mean dataset classification times for each algorithm and attribute set level tested. The fastest classification time is underlined in the table for each attribute set level. Again, Naïve Bayes had the fastest dataset classification times for all three attribute set levels: 13.873 seconds with the full attribute set, 6.076 seconds with the wrapper attribute subset, and 7.571 seconds with the filter attribute subset. Furthermore, NBTree had the slowest dataset classification times for all three attribute set levels. BayesNet had a dataset classification time of 68.994 seconds with the full attribute set and 17.301 with the filter attribute set.

The dataset classification time is a more critical performance metric since it demonstrates how quickly the algorithm can classify continuous network traffic. Algorithms with faster dataset classification times are more feasible for implementation into a real-world device. Similar to the model build time, the dataset classification times improved significantly for all four algorithms when the wrapper and filter attribute subsets are used.

4.3.3 Attribute Set Selection.

The dataflow classification results when using the full attribute list had a peak TPR of .9933 with the BayesNet algorithm. However, BayesNet's mean model build time was 10.906 seconds and dataset classification time was 68.994 seconds when using the full attribute set. The accuracy results when using the filter attribute subset showed a peak TPR of .9935 with BayesNet while its model build time dropped to 1.799 seconds and dataset classification time to 17.301 seconds.

Note that the dataset classification times and model build times were faster for all four algorithms for the wrapper attribute as compared to the full attribute set. The BayesNet algorithm's dataset classification time and model build time, however, increased for the filter attribute subset as compared to the wrapper attribute subset. The time increase is a result of the BayesNet wrapper subset only containing three attributes, as compared to the five attributes in the filter attribute set; the other three algorithms contained more than five attributes in their respective wrapper subsets. The results demonstrate that using the filter attribute subset meets the required SCADA dataflow classification accuracy while reducing the timing overhead of the full attribute set.

4.4 Summary

The first research goal of demonstrating the ability to identify SCADA network traffic in a mixed network traffic trace, obtaining at least a .99 TPR, is achieved with the BayesNet algorithm. Naïve Bayes, NBTree, and J4.8 Decision Tree demonstrate SCADA network traffic identification results of greater than .96. While not meeting the goal of a .99 TPR, the results offer promising indicators for using ML techniques in SCADA traffic classification.

The second research goal of identifying an optimal subset of attributes while maintaining at least a .99 TPR is also met with BayesNet's peak TPR of .9935 when using the filter attribute subset. The top five ranked attributes used as the filter attribute subset include attributes from all three traffic behavior categories. This reinforces the distinction of SCADA network traffic from traditional ICT network traffic based on packet timing, packet size, and data throughput.

Notional evaluation is performed on the classification model build and dataset classification times for each algorithm and attribute set level. Naïve Bayes demonstrated the fastest classification model build times for all three attribute set levels, with mean times of .229 seconds with the filter attribute subset and .324 seconds with the wrapper attribute subset. Naïve Bayes also had the fastest dataset classification times for all three attribute set levels, with mean times of 6.076 seconds with the filter attribute subset and 7.571 seconds with the wrapper attribute subset. The model build times and dataset classification times improved significantly when the wrapper and filter subsets were used with all four ML algorithms.

V. Conclusions

This chapter summarizes the results of the research. Section 5.1 discusses the conclusions based on the results in Chapter 4. Section 5.2 lists the contributions of the research, and Section 5.3 describes recommendations for future work.

5.1 Conclusions

Four supervised ML algorithms are tested on their ability to classify SCADA device dataflows within a mixed traffic network: Naïve Bayes, NBTree, BayesNet, and J4.8 Decision Tree. The mixed traffic trace used to test the four algorithms is generated using traces collected from real-world SCADA and traditional ICT networks. Twenty-four attributes based on packet timing, packet size, and data throughput are calculated from each dataflow in the mixed traffic trace.

Each algorithm is tested using three attribute set levels: the full attribute set, the wrapper function subset, and the filter function subset. The first goal of the research was to demonstrate that a TPR of at least .99 is feasible for identifying SCADA network traffic within a mixed network traffic trace; therefore, an algorithm is considered effective if it has a TPR of at least .99. It is also important to not misclassify ICT dataflows as SCADA; therefore, an FPR of < .05 is desirable. The second goal was to find an optimal subset of attributes that maintain the .99 TPR for SCADA network traffic identification. BayesNet, when using the full attribute set and the filter attribute subset, achieved the goal with TPRs of .9933 and .9935, respectively. Furthermore, BayesNet's FPR was less than .01 when using both attributes sets. Therefore, both research goals were achieved with BayesNet using the filter attribute subset.

The top five ranked attributes used as the filter attribute subset included attributes from all three traffic behavior categories. This reinforced the distinct behaviors of SCADA network traffic from ICT network traffic based on packet timing, packet size, and data throughput.

5.2 Contributions

The research presented four supervised ML algorithms for use in identifying SCADA network traffic in a mixed network traffic trace, and demonstrated that ML algorithms can achieve an acceptable TPR in this context. It also presented two optimal attribute subsets for identifying SCADA network traffic.

This research also analyzed the time to build the classification model and to classify the dataset for each algorithm using the full list of 24 flow-based attributes and the two optimal attribute subsets. As mixed traffic networks in the corporate environment become more commonplace, the ability to differentiate between SCADA network traffic and traditional ICT network traffic is critical to security [11, 48].

This work furthers the current research in the field by contributing the proven capability to identify SCADA network traffic using an optimal subset of flow-based attributes and ML techniques to achieve > .99 TPR for a given network. To date, using ML algorithms and flow-based attributes is a novel approach for SCADA network traffic identification. Typical traffic classification techniques have used packet information such as port, protocol, IP address and payload content to classify network traffic (e.g., [9, 56]). The ability to identify a SCADA device on a mixed traffic network without prior knowledge of protocol, port, or IP address is necessary as SCADA devices tend to use proprietary protocols and non-standard ports. Instances, such as when Google's building management system (BMS) was accessed and administrator password retrieved from the Internet [42], reveal that many times asset owners are not only unaware that SCADA devices are on their networks but these devices are also connected to the Internet. As mixed traffic networks in the corporate environment become more commonplace, the ability to identify SCADA device traffic using traditional means such as port, protocol and

IP address, as well as non-traditional means such as flow-based statistics, is necessary [11].

5.3 Future Work

The items listed below are suggestions for future work that could expand on the research presented here.

- **Use SCADA and Traditional ICT Device Types as Labels**

  This research only classified dataflows as originating from either a SCADA or ICT device. The dataset created for this research contained dataflows from two SCADA facilities and a variety of traditional ICT devices. Classifying or labeling dataflows based on the type of device (i.e., PLC, building automation system (BAS), mail server, or print server) could provide further insight into those specific device dataflow behaviors or signatures.

- **Test Algorithms On Different Network Traffic**

  As discussed in Chapter 3, performance is expected to change for different networks. Extending this research to other ICT and SCADA network traffic traces provides further insight into classification and may help further refine results.

- **Test Accuracy Stability Over Time**

  While SCADA networks tend to have a static topology, traditional ICT network topologies change drastically over time as devices are constantly added and removed. This research is intended for mixed traffic networks, where the topology is dynamic. By creating datasets from the same three networks used in this research with traces taken at a later date, the classification accuracy of the algorithms can re-examined. By using the classification models created during this research on the new datasets, it can be determined whether the classifiers require updating over time.

- **Use New Flow-Based Statistics**

  The 24 flow-based statistical attributes yielded acceptable classification accuracy results; however, the optimal subset of attributes for this dataset found packet timing to be the most influential traffic behavior category for distinguishing SCADA traffic dataflows. Further research exploring this behavior and testing other attribute categories may achieve higher classification accuracies.

- **Use Different Supervised ML Algorithms**

  This research used four supervised ML algorithms. The algorithms were chosen based on previous research success for traffic classification problems. There are numerous supervised ML algorithms available in Weka that may yield higher classification accuracies.

- **Test With Various Dataflow Mixes**

  This research used a dataset which contained 116,527 mixed dataflows. Of those, only 1,736 dataflows were SCADA dataflows, while the other 114,791 were traditional ICT dataflows. This mixture is intended to represent a typical mixed traffic network with significantly more traditional device traffic than SCADA device. Classification accuracy results may vary depending on the dataflow mixture within the dataset. Testing the algorithms with various dataflow mixtures may yield different classification accuracy results.

- **Implement On A Real-World Mixed Network**

  This research demonstrated that SCADA dataflow classification is feasible using ML with a mixed traffic trace created from real-world network traffic. The next phase of this research includes extending the implementation to an operational, real-time network.

5.4 Summary

SCADA devices are being connected to corporate networks to provide remote control and monitoring, usage reporting and automated billing capabilities. As such, network traffic classification techniques specific to SCADA device traffic are necessary. This research presents a novel technique of utilizing supervised ML algorithms and flow-based statistics to accurately classify SCADA traffic in a mixed traffic network. By utilizing this technique, SCADA devices can be identified without prior knowledge of port, protocol, IP address or payload content information.

Bibliography

[1] Barbosa, R., R. Sadre, and A. Pras. “Difficulties in modeling SCADA traffic: a comparative analysis”. *Passive and Active Measurement*, 126–135. Springer, 2012.

[2] Barbosa, R., R. Sadre, and A. Pras. “A first look into SCADA network traffic”. *Network Operations and Management Symposium (NOMS), 2012 IEEE*, 518–521. IEEE, 2012.

[3] Barto, W. *Classification of Encrypted Web Traffic Using Machine Learning*. Ph.D. thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, OH, 2013.

[4] Borders, K. and A. Prakash. “Quantifying information leaks in outbound web traffic”. *Security and Privacy, 2009 30th IEEE Symposium on*, 129–140. IEEE, 2009.

[5] Bouckaert, R. *Bayesian network classifiers in weka*. Department of Computer Science, University of Waikato, 2004.

[6] Boyer, S. *SCADA: supervisory control and data acquisition*. International Society of Automation, 2009.

[7] Byres, E. “Project SHINE: 1,000,000 Internet Connected SCADA and ICS Systems and Counting”, september 2013. URL http://www.tofinosecurity.com/blog/project-shine-1000000-internet-connected-scada-and-ics-systems-and-counting.

[8] Chaudhary, A. and A. Sardana. “Software Based Implementation Methodologies for Deep Packet Inspection”. *Information Science and Applications (ICISA), 2011 International Conference on*, 1–10. IEEE, 2011.

[9] Cheung, S., B. Dutertre, M. Fong, U. Lindqvist, K. Skinner, and A. Valdes. “Using model-based intrusion detection for SCADA networks”. *Proceedings of the SCADA Security Scientific Symposium*, 1–12. 2007.

[10] Claffy, K. *Internet traffic characterization*. Ph.D. thesis, University of California, San Diego, 1994.

[11] Department of Energy. “21 Steps to Improve Cyber Security of SCADA Networks”. URL http://www.oe.netl.doe.gov/docs/prepare/21stepsbooklet.pdf.

[12] Dewes, C., A. Wichmann, and A. Feldmann. “An analysis of Internet chat systems”. *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*, 51–64. ACM, 2003.

[13] Erman, J., M. Arlitt, and A. Mahanti. “Traffic classification using clustering algorithms”. *Proceedings of the 2006 SIGCOMM workshop on Mining network data*, 281–286. ACM, 2006.

[14] Esposito, R. “Hackers Penetrate Water System Computers”, October 2006. URL http://abcnews.go.com/blogs/headlines/2006/10/hackers_penetra/.

[15] Falliere, N., L. Murchu, and E. Chien. *W32.Stuxnet.Dossier*. Symantic Corporation, Cupertino, CA, rep. ver. 1.4 edition, February 2011.

[16] Gertz, B. “The Cyber-Dam Breaks”, May 2013. URL http://freebeacon.com/the-cyber-dam-breaks/.

[17] Gronewaldstrae, T. and F. Thiessenhusen. “SCADA systems are used in process data networks”. URL http://www.all-about-security.de/kolumnen/frage-des-monats/browse/6/artikel/7973-admeritia-scada-systeme-kommen-in-sogenannten-prozessdatenn/.

[18] Guo, G., S. Li, and K. Chan. “Face recognition by support vector machines”. *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, 196–201. IEEE, 2000.

[19] Hall, M., E. Frank, G.Holmes, B. Pfahringer, P. Reutemann, and I. Witten. “The WEKA Data Mining Software: An Update”, 2009. URL http://www.cs.waikato.ac.nz/ml/weka/).

[20] Han, J., M. Kamber, and J. Pei. *Data mining: concepts and techniques*. Morgan kaufmann, 2006.

[21] ICS-CERT. “ICS-CERT Monthly Monitor: April-June 2013”, 2013. URL http://ics-cert.us-cert.gov/monitors/ICS-MM201306.

[22] ICS-CERT. “Industrial Control System Cyber Emergency Response Team”, September 2013. URL http://ics-cert.us-cert.gov/.

[23] Japkowicz, N. and M. Shah. *Evaluating learning algorithms: a classification perspective*. Cambridge University Press, 2011.

[24] Kaelbling, L., M. Littman, and A. Moore. “Reinforcement learning: A survey”. *arXiv preprint cs/9605103*, 1996.

[25] Kohavi, R. “A study of cross-validation and bootstrap for accuracy estimation and model selection”. *IJCAI*, volume 14, 1137–1145. 1995.

[26] Lang, T., G. Armitage, P. Branch, and H. Choo. “A synthetic traffic model for Half-Life”. *Australian Telecommunications Networks & Applications Conference*, volume 2003. 2003.

[27] Lang, T., P. Branch, and G. Armitage. "A synthetic traffic model for Quake3". *Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology*, 233–238. ACM, 2004.

[28] Li, W. and A. Moore. "A Machine Learning Approach for Efficient Traffic Classification". *Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2007. MASCOTS '07. 15th International Symposium on*, 310–317. 2007.

[29] Li, Z., R. Yuan, and X. Guan. "Accurate classification of the internet traffic based on the svm method". *Communications, 2007. ICC'07. IEEE International Conference on*, 1373–1378. IEEE, 2007.

[30] Ling, C., J. Huang, and H. Zhang. "AUC: a statistically consistent and more discriminating measure than accuracy". *IJCAI*, volume 3, 519–524. 2003.

[31] Mahmood, A., C. Leckie, J. Hu, Z. Tari, and M. Atiquzzaman. "Network traffic analysis and SCADA security". *Handbook of Information and Communication Security*, 383–405. Springer, 2010.

[32] Mitchell, T. "Machine learning and data mining". *Communications of the ACM*, 42(11):30–36, 1999.

[33] Moore, A. and D. Zuev. "Internet traffic classification using Bayesian analysis techniques". *Performance evaluation review*, volume 33, 50–60. Association for Computing Machinery, 2005. ISBN 0163-59990163-5999.

[34] Moore, A., D. Zuev, and M. Crogan. *Discriminators for use in flow-based classification*. Technical Report RR-05-13, Department of Computer Science, Queen Mary, University of London, August 2005 2005.

[35] Naraine, R. "Shodan search exposes insecure SCADA systems", November 2010. URL http://www.zdnet.com/blog/security/ shodan-search-exposes-insecure-scada-systems/7611.

[36] NERC. "Critical Infrastructure Protection Committee", 2013. URL http://www.nerc.com/comm/CIPC/Pages/default.aspx.

[37] Nguyen, T. and G. Armitage. "A survey of techniques for internet traffic classification using machine learning". *Communications Surveys and Tutorials, IEEE*, 10(4):56–76, 2008.

[38] OWASP. "SQL Injection", April 2013. URL https://www.owasp.org/index.php/SQL_Injection.

[39] Pantel, P. and D. Lin. "Spamcop: A spam classification & organization program". *Proceedings of AAAI-98 Workshop on Learning for Text Categorization*, 95–98. 1998.

[40] Paxson, V. “Empirically derived analytic models of wide-area TCP connections”. *IEEE/ACM Transactions on Networking (TON)*, 2(4):316–336, 1994.

[41] Powner, D. and K. Rhodes. *Critical Infrastructure Protection: Multiple Efforts to Secure Control Systems are Under Way, but Challenges Remain*. Technical Report rep. GAO-07-1036, 2007.

[42] Price, B. “Google Building Management System Hack Highlights SCADA Security Challenges”, May 2013. URL http://www.darkreading.com/vulnerability/google-building-management-system-hack-h/240154553.

[43] Quinlan, J. *C4.5 Programs for Machine Learning*. Morgan Kaufman, San Mateo, California, 1993.

[44] Shams, R. “Weka Tutorial 09: Feature Selection (Wrapper)”, 2013. URL https://www.youtube.com/watch?v=x5wa1w-BpRE.

[45] Shams, R. “Weka Tutorial 10: Feature Selection (Filter)”, 2013. URL https://www.youtube.com/watch?v=UOadhDKRbPM.

[46] Shams, R. “Weka Tutorial 11: How to get the Folds from Cross Validation Sets”, 2013. URL https://www.youtube.com/watch?v=VU315kgS7RM.

[47] Smith, B. *Kernel Extended Real-Valued Negative Selection Algorithm (KERNSA)*. Ph.D. thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, OH, 2013.

[48] Stouffer, K., J. Falco, and K. Scarfone. *Guide to Industrial Systems (ICS) Security*. Technical report, National Institute of Standards and Technology, june 2011.

[49] Stuart, J. and P. Norvig. *Artificial Intelligence: a modern approach*. Prentice Hall, Upper Saddle River, NJ, third edition edition, 2010.

[50] Symantec. “Waterhole Attack”, September 2012. URL http://www.slideshare.net/symantec/waterhole-attack.

[51] Symantec. “Spear Phishing: Scam, Not Sport”, 2013. URL http://us.norton.com/spear-phishing-scam-not-sport/article.

[52] Tcpdump/Libpcap. “Tcpdump & Libpcap”, 2010. URL http://www.wireshark.org/.

[53] U.S. Department of Homeland Security. “Homeland Security Presidential Directive 7”, December 2003. URL https://www.dhs.gov/homeland-security-presidential-directive-7.

[54] U.S. Department of Homeland Security. “National Infrastructure Protection Plan”, 2013. URL https://www.dhs.gov/national-infrastructure-protection-plan.

[55] U.S. Department of Homeland Security. "What is Critical Infrastructure?", July 2013. URL http://www.dhs.gov/what-critical-infrastructure.

[56] Valdes, Alfonso and Steven Cheung. "Communication pattern anomaly detection in process control systems". *Technologies for Homeland Security, 2009. HST'09. IEEE Conference on*, 22–29. IEEE, 2009.

[57] Williams, N. "A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification". *Computer communication review*, 36(5):7–15, 2006.

[58] Wireshark. "Wireshark", September 2013. URL http://www.wireshark.org/.

[59] Witten, I. and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.

[60] Witten, I., E. Frank, and M. Hall. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, Burlington, MA, 3rd edition, 2011.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704–0188*

1. REPORT DATE *(DD–MM–YYYY)* 27–03–2014	**2. REPORT TYPE** Master's Thesis	**3. DATES COVERED** *(From — To)* Aug 2012–Mar 2014

4. TITLE AND SUBTITLE Behavioral profiling of SCADA network traffic using machine learning algorithms	**5a. CONTRACT NUMBER** **5b. GRANT NUMBER** **5c. PROGRAM ELEMENT NUMBER**
6. AUTHOR(S) Werling, Jessica R., Captain, USAF	**5d. PROJECT NUMBER** **5e. TASK NUMBER** **5f. WORK UNIT NUMBER**
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 Hobson Way WPAFB, OH 45433-7765	**8. PERFORMING ORGANIZATION REPORT NUMBER** AFIT-ENG-14-M-81
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Department of Homeland Security ICS-CERT POC: Nick Carr Nicholas.Carr@HQ.DHS.GOV 245 Murray Lane SW Bldg 410, Mail Stop 635 Washington, DC 20528	**10. SPONSOR/MONITOR'S ACRONYM(S)** DHS ICS-CERT **11. SPONSOR/MONITOR'S REPORT NUMBER(S)**

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

13. SUPPLEMENTARY NOTES

This work is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT

Mixed traffic networks containing both traditional ICT network traffic and SCADA network traffic are more commonplace now due to the desire for remote control and monitoring of industrial processes. The ability to identify SCADA devices on a mixed traffic network with zero prior knowledge, such as port, protocol or IP address, is desirable since SCADA devices are communicating over corporate networks but typically use non-standard ports and proprietary protocols.
Four supervised ML algorithms are tested on a mixed traffic dataset containing 116,527 dataflows from both SCADA and traditional ICT networks: Naïve Bayes, NBTree, BayesNet, and J4.8. Using packet timing, packet size and data throughput as traffic behavior categories, this research calculates 24 attributes from each device dataflow. All four algorithms are tested with three attribute subsets: a full set and two reduced attribute subsets.
The attributes and ML algorithms chosen for experimentation successfully demonstrate that a TPR of .9935 for SCADA network traffic is feasible on a given network. It also successfully identifies an optimal attribute subset, while maintaining at least a .99 TPR. The optimal attribute subset provides the SCADA network traffic behaviors that most effectively differentiating them from traditional ICT network traffic.

15. SUBJECT TERMS

Supervisory Control and Data Acquisition, Network Analysis, Industrial Control Systems, Machine Learning, Network Traffic Behavior, Network Traffic Characterization, Dataflow Characterization, Supervised Learning Algorithms

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Maj Jonathan W. Butts, PhD
					19b. TELEPHONE NUMBER *(include area code)*
U	U	U	UU	86	(937) 255-3636 ext.4332, jonathan.butts@afit.edu